ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ & ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ

Διπλωματική Εργασία

Σχεδιασμός και Ανάπτυξη Έξυπνων Συμβολαίων και Κατανεμημένων Εφαρμογών σε Ethereum Blockchain

Φοιτητής: Καρσλίδης Δημήτριος
ΑΜ: 50106729

Επιβλέπων Καθηγητής

Κόγιας Δημήτριος
Ακαδημαϊκός Υπότροφος/ ΕΣΠΑ

ΑΘΗΝΑ-ΑΙΓΑΛΕΩ, Απρίλιος 2021

UNIVERSITY OF WEST ATTICA
FACULTY OF ENGINEERING
DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING

Diploma Thesis

Design and Development of Smart Contracts and Distributed Applications (D-apps) in Ethereum Network

Student: Karslidis Dimitrios
Registration Number: 50106729

Supervisor

Kogias Dimitrios
Academic Scholar/ ESPA

ATHENS-EGALEO, April 2021

Η Διπλωματική Εργασία έγινε αποδεκτή και βαθμολογήθηκε από την εξής τριμελή επιτροπή:

(Ονοματεπώνυμο), (βαθμίδα)	(Ονοματεπώνυμο), (βαθμίδα)	(Ονοματεπώνυμο), (βαθμίδα)
ΔΗΜΗΤΡΙΟΣ ΚΟΓΙΑΣ (Υπογραφή)	 Digitally signed by Charalampos Patrikakis Date: 2021.05.24 10:45:11 +03'00' (Υπογραφή)	Periklis Papadopoulos Digitally signed by Periklis Papadopoulos Date: 2021.05.25 09:47:26 +03'00' (Υπογραφή)

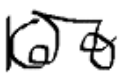
ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Ο/η κάτωθι υπογεγραμμένος/η Καρσλίδης Δημήτριος του Αρτούρ, με αριθμό μητρώου 50106729 φοιτητής του Πανεπιστημίου Δυτικής Αττικής της Σχολής ΜΗΧΑΝΙΚΩΝ του Τμήματος ΗΛΕΚΤΡΟΛΟΓΩΝ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ,

δηλώνω υπεύθυνα ότι:

«Είμαι συγγραφέας αυτής της διπλωματικής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος.
Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του διπλώματός μου.
Δεν επιθυμώ την απαγόρευση πρόσβασης στο πλήρες κείμενο της εργασίας μου.»

Ο/Η Δηλών/ούσα
Καρσλίδης Δημήτριος



(Υπογραφή φοιτητή/ήτριας)

ΑΦΙΕΡΩΣΗ

Αυτή η διπλωματική είναι αφιερωμένη στην οικογένεια μου, για την βοήθεια τους και την στήριξη τους όλα αυτά τα χρόνια.

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω τον Επιβλέποντα καθηγητή Κόγια Δημήτριο για την πρόταση να αναλάβω έναν τόσο ενδιαφέρον θέμα και για την βοήθεια που μου παρείχε για την εκπόνηση αυτής της διπλωματικής καθώς επίσης και τον συμφοιτητή και φίλο Παναγιώτη Κορομηλά για την συνεργασία και την αλληλοβοήθεια καθ' όλη την διάρκεια των σπουδών.

Περίληψη

Η διπλωματική εργασία έχει σαν σκοπό να μελετήσει τον τρόπο λειτουργίας, την σχεδίαση και την υλοποίηση μιας αποκεντρικοποιημένης εφαρμογής με την βοήθεια της δημοφιλούς πλατφόρμας Ethereum, η οποία είναι μία υλοποίηση Blockchain. Στο θεωρητικό μέρος της διπλωματικής, αρχικά θα αναλυθεί πως λειτουργούν τα διαφορά είδη αρχιτεκτονικών συστημάτων και τις διαφορές τους, στην συνέχεια θα δούμε πως λειτουργεί η τεχνολογία blockchain όσον αφορά τα μπλοκ και το πως επιλέγονται και τοποθετούνται αυτά στην αλυσίδα του blockchain, καθώς και τα μειονεκτήματα και τα πλεονεκτήματα αυτής της τεχνολογίας. Στην συνέχεια θα επικεντρωθούμε στο blockchain του Ethereum, αναφέροντας γιατί υπερτερεί έναντι άλλων blockchain όπως του Bitcoin, πως λειτουργεί χρησιμοποιώντας το Ethereum Virtual Machine και τι είναι τα έξυπνα συμβόλαια που είναι τα κύρια χαρακτηριστικά του. Θα δούμε τα τοκεν που επιτρέπουν διάφορες λειτουργίες όπως οι συναλλαγές για περιουσιακά στοιχεία και τέλος θα πούμε τι είναι το gas, το οποίο αποτρέπει την κακόβουλη κατάχρηση του δικτύου.

Στο πρακτικό κομμάτι θα αναφερθούν συνοπτικά τα εργαλεία που χρησιμοποιήθηκαν για το κομμάτι της υλοποίησης και τι είναι οι αποκεντρικοποιημένες εφαρμογές (D-Apps). Τέλος, θα παρουσιαστεί η εφαρμογή D-App που υλοποιήθηκε στο πλαίσιο της διπλωματικής και οι λειτουργίες της χρησιμοποιώντας τις τεχνολογίες που αναφέρθηκαν στο θεωρητικό κομμάτι.

Λέξεις – κλειδιά

Αποκεντρικοποιημένες εφαρμογές, Έξυπνα συμβόλαια, Ethereum blockchain, Cryptocurrency, Κατανεμημένο εδάφιο.

Abstract

The aim of this thesis is to study how decentralized applications work and the design and development of one, with the help of the popular platform of Ethereum's Blockchain.On the theoretical part, first, the various system architectures will be analyzed and their differences. Then we will see how the blockchain technology works, the blocks and how they are getting chose and added on the chain and also the pros and cons of this technology. Afterwards we will focus on the Ethereum blockchain and what advantages it has over other blockchain implementations like Bitcoin. We will present how it works using the Ethereum Virtual Machine (EVM) and what is a smart contract. We will also, discuss about the main features of Ethereum and we will also, present the use of tokens, which are used to add some additional functionalities like transactions for assets. Finally, we will explain what is gas and how it impedes the malicious use of the EVM.
On the practical part we will do a quick reference of the tools which are commonly used for the development of a DApp while, at the same time explain what a D-App is and lastly, we will present the D-App that has been developed for this thesis and its functionalities by using the technologies discussed on the theoretical part.

Keywords

Decentrilized applications, Smart contracts, Ethereum blockchain, Cryptocurrency, Distributed ledger.

Περιεχόμενα

Κατάλογος Εικόνων

1 ΕΙΣΑΓΩΓΗ

Με την ανακάλυψη του int“Στο διαδίκτυο, κάνεις δεν ξέρει εάν είσαι σκύλος” -Peter Steiner

“Στο blockchain, κανείς δεν ξέρεις αν είσαι ψυγείο” -Richard Gendal Brown

ernet και την δυνατότητα των ηλεκτρονικών συναλλαγών, όταν γίνεται κάποια συναλλαγή, αυτή καταγράφεται σε μία βάση δεδομένων, η οποία ανήκει σε κάποια τράπεζα. Οι άνθρωποι εμπιστεύονται τις τράπεζες ότι δεν θα τους κλέψουν τα λεφτά, ότι κατά την πραγματοποίηση κάποιας ηλεκτρονικής συναλλαγής θα αφαιρεθεί ή θα πιστωθεί το σωστό ποσό, καθώς επίσης ότι θα μπορέσουν να χρησιμοποιήσουν αυτά τα λεφτά στην συνέχεια για άλλες συναλλαγές. Τις εμπιστεύονται επειδή αυτές ελέγχονται από την κυβέρνηση.

Σε χώρες που υπάρχει έλλειψη εμπιστοσύνης ως προς τις εταιρίες ή την κυβέρνηση, οι συναλλαγές είναι δύσκολες, διότι, αν βάλουν τα λεφτά στην τράπεζα ρισκάρουν να τα κλέψει η τράπεζα ή κατά την διάρκεια μιας μεγάλης συναλλαγής όπως η αγορά σπιτιού, το άλλο εμπλεκόμενο στην συναλλαγή μέλος να μη τηρήσει την συμφωνία.

Τι ενναλακτικές έχουν οι πολίτες εκτός από το να καταθέσουν τα λεφτά τους στις τράπεζες που θεωρούσαν αξιόπιστες; Σε περίπτωση πυρκαγιάς ή διάρρηξης υπάρχει ο κίνδυνος να χαθούν όλα τα χρήματα. Αν υπάρχει ο κίνδυνος να καταρρεύσουν οι τράπεζες και οι κυβερνήσεις να παγώσουν λογαριασμούς σε Αμερική και Ευρώπη, πώς μπορούν οι πολίτες σε λιγότερο ανεπτυγμένες χώρες να τις εμπιστευτούν; Η απάντηση είναι πώς δεν μπορούν.

Οι κεντρικοποιημένες βάσεις δεδομένων και θεσμοί λειτουργούν όταν υπάρχει εμπιστοσύνη στον νόμο, στους ελέγχους, στην κυβέρνηση, στις τράπεζες και στους ανθρώπους, αλλά ακόμα και αν όλοι αυτοί οι παράγοντες είναι έμπιστη, μπορεί ακόμα να προδοθεί αυτή η εμπιστοσύνη με αποτέλεσμα την απώλεια χρημάτων η περιουσιακών στοιχείων.

Μία αποκεντρικοποιημένη βάση δεδομένων φτιαγμένη στο blockchain αφαιρεί την ανάγκη κεντρικοποιημένων θεσμών. Όλοι στην αλυσίδα μπορούν να δουν και να επαληθεύσουν συναλλαγές δημιουργώντας διαφάνεια και εμπιστοσύνη χωρίς την ανάγκη μεσολάβησης από τρίτους.[12]

Τέτοιου είδους blockchain μας επιτρέπουν όμως μόνο την δυνατότητα χρηματικών συναλλαγών και δεν είναι δυνατή η χρήση του σε περιπτώσεις, όπως για παράδειγμα, γίνεται μία ηλεκτρονική ψηφοφορία και δεν υπάρχει εμπιστοσύνη στον διοργανωτή, ή η εύκολη και αυτοματοποιημένη μισθοδοσία εργαζόμενων και άλλων διάφορων εφαρμογών που απαιτούν πιο σύνθετη προγραμματιστική λογική και αποκεντρικοποίηση από θεσμούς.

Το blockchain του Ethereum όμως επιτρέπει επιπρόσθετα την δημιουργία έξυπνων συμβολαίων και λόγω της πληρότητας Turing, είναι δυνατή η δημιουργία απλών έως και πιο σύνθετων αποκεντρικοποιημένων εφαρμογών, όπως τα παραπάνω παραδείγματα, όπου είναι επιθυμητή η ανεξαρτησία από κάποιον κεντρικό φορέα ή εταιρία, μόνο και μόνο γράφοντας την λογική της εφαρμογής σε μορφή κώδικα.

1.1 Αντικείμενο της διπλωματικής εργασίας

Το θέμα της διπλωματικής είναι η σχεδίαση και η ανάπτυξη μίας αποκεντρικοποιημένης εφαρμογής. Αυτές οι εφαρμογές εκμεταλλεύονται την τεχνολογία blockchain του Ethereum για την αποκεντρικοποίηση τους και κάνουν χρήση έξυπνων συμβολαίων για την ενίσχυση της λειτουργικότητας τους. Πιο συγκεκριμένα θα υλοποιηθεί ένα text based παιχνίδι ρόλων, στο οποίο ο χρήστης κάθε φορά που νικάει μια μάχη θα έχει την πιθανότητα να αποκτήσει ένα μοναδικό cryptocollectible το οποίο μπορεί να πουλήσει για οικονομικό όφελος ή και να αγοράσει από άλλους χρήστες για να μεγαλώσει την συλλογή του. Αυτές οι εφαρμογές λόγω της αποκεντρικοποιημένης φύσης τους επιτρέπουν γρήγορες, φθηνές και χωρίς μεσάζοντα συναλλαγές.

1.2 Σκοπός και στόχοι

Σκοπός της εργασίας είναι η υλοποίηση μίας αποκεντρικοποιημένης εφαρμογές, η οποία θα έχει διάφορες λειτουργίες μέσω της χρήσης έξυπνων συμβολαίων στο blockchain του Ethereum, το οποίο επιλέχθηκε έναντι άλλων blockchain, λόγω της πληρότητας Turing, το οποίο είναι σημαντικό επειδή δίνει την δυνατότητα υλοποίησης εφαρμογών. Σκοπός της εφαρμογής είναι η πραγματοποίηση γρήγορων και χαμηλού κόστους συναλλαγών χωρίς την ανάγκη μεσαζόντων.

1.3 Μεθοδολογία

Η μεθοδολογία που ακολουθήθηκε είναι η εξής :

- Έγινε έρευνα για τις διάφορες τεχνολογίες που χρειάζονται για την υλοποίηση.
- Έγινε εκμάθηση της γλώσσας solidity για τη δημιουργία και συγγραφή των έξυπνων συμβολαίων.
- Υλοποιήθηκαν τα έξυπνα συμβόλαια τοπικά, τα οποία είναι συμβατά με την πλατφόρμα του Ethereum Blockchain, και πραγματοποιήθηκε η αποσφαλμάτωση τους.
- Υλοποιήθηκε το front end της εφαρμογής και η διασύνδεση του με το Blockchain δίκτυο.

1.4 Καινοτομία

Στοιχεία της διπλωματικής εργασίας που είναι καινοτομικά / πρωτότυπα

Τα καινοτόμα στοιχεία της διπλωματικής είναι η τεχνολογία blockchain με την οποία είναι δυνατές οι συναλλαγές χωρίς μεσάζοντες και χωρίς την ανάγκη εμπιστοσύνης μεταξύ των εμπλεκόμενων, το Ethereum όπου σε συνδυασμό με τα έξυπνα συμβόλαια επιτρέπουν την δημιουργία εφαρμογών χρησιμοποιώντας τα οφέλη του blockchain και τα crypto collectibles τα οποία είναι ψηφιακά συλλεκτικά αντικείμενα .

1.5 Δομή

Η δομή της διπλωματικής έχει ως εξής :

Στο πρώτο κεφάλαιο γίνεται μία εισαγωγή στην ιδέα της Διπλωματικής αυτής. Στο δεύτερο κεφάλαιο αναφέρονται οι αρχιτεκτονικές συστημάτων και οι διαφορές τους, στην συνέχεια επικεντρωνόμαστε στα κατανεμημένα συστήματα, έπειτα γίνεται μια σύντομη αναφορά στην ιστορία του blockchain και του Ethereum και στο πως λειτουργούν τα κλειδιά και οι διευθύνσεις επικεντρώνοντας σε αυτό του Ethereum, μετά από αυτό αναλύουμε τα μπλοκ, όπως η επικεφαλίδα του και τα αναγνωριστικά του, και την διαδικασία που ακολουθείται για την δημιουργία και την προσθήκη τους στην αλυσίδα και τέλος αναφέρονται τα πλεονεκτήματα, τα μειονεκτήματα και οι απειλές του blockchain.

Στο τρίτο κεφαλαίο αναλύουμε το blockchain του Ethereum και τα πλεονεκτήματα του έναντι σε άλλα blockchain, εξηγείται τι είναι το Ethereum Virtual Machine και τι μας προσφέρει, έπειτα αναφέρονται διαφορές πληροφορίες για τα έξυπνα συμβόλαια, όπως το τι είναι και καλές πρακτικές που πρέπει να ακολουθούνται, ακόμα γίνεται αναφορά στα τοκεν και τις χρήσεις τους και τέλος αναφέρεται συνοπτικά ο ρόλος του gas στο Ethereum.

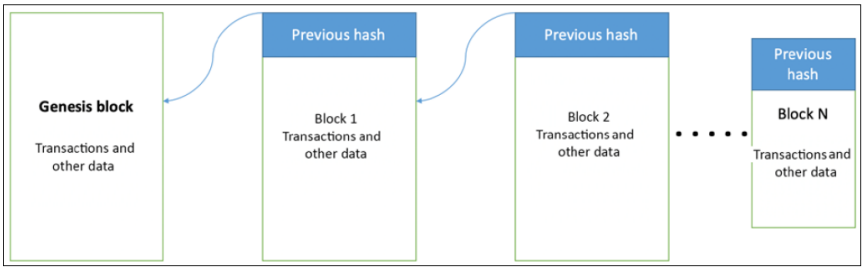
Στο τέταρτο και τελευταίο κεφάλαιο γίνεται σύντομη αναφορά στα εργαλεία που χρησιμοποιήθηκαν για την υλοποίηση του DApp καθώς και το τι είναι και τέλος γίνεται η παρουσίαση του DApp και του πως λειτουργεί.

2 ΚΕΦΑΛΑΙΟ 2ο : Blockchain

Για να καταλάβουμε καλύτερα το blockchain ας δούμε πρώτα τί είναι κατανεμημένο εδάφιο(Distributed ledger). Ένα κατανεμημένο εδάφιο στην πιο απλή μορφή του είναι μία βάση δεδομένων την οποία την κατέχει και μπορεί να την ενημερώσει ανεξάρτητα κάθε κόμβος σέ ένα δίκτυο. Η ιδιαιτερότητα αυτού του εδάφιου είναι πώς τις καταγραφές δεν τις διαχειρίζεται ένα αρμόδιο άτομο αλλά όλοι όσοι συμμετέχουν στο δίκτυο. Κάθε κόμβος επεξεργάζεται κάθε συναλλαγή, φτάνοντας σε ένα συμπέρασμα και ύστερα ψηφίζοντας σε αυτό το συμπέρασμα έτσι ώστε να σιγουρέψει ότι η πλειοψηφία θα συμφωνήσει με αυτό. Όλοι οι κόμβοι κατέχουν ένα πανομοιότυπο εδάφιο και μόλις υπάρξει συναίνεση, το εδάφιο όλων των κόμβων ενημερώνεται.[7]

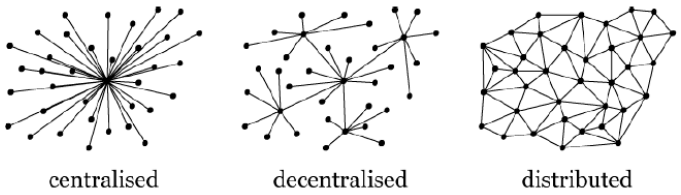
Η κύρια διαφορά μίας βάσης δεδομένων και ενός εδάφιου, είναι ότι σε μία βάση μπορούμε να προσθέσουμε, να τροποποιήσουμε και να διαγράψουμε συναλλαγές, όμως σε ένα εδάφιο είναι δυνατή μόνο η προσθήκη καινούργιων συναλλαγών.[27]

Το blockchain είναι μία δομή δεδομένων η οποία χρησιμοποιείται για την δημιουργία ενός αποκεντρικοποιημένου εδάφιου. Το blockchain αποτελείται από μπλοκ συνδεδεμένα σειριακά με ψηφιακό τρόπο, όπως μπορούμε δούμε στην εικόνα 2.1, και κάθε μπλοκ περιέχει διάφορες πληροφορίες που θα δούμε στην υπό ενότητα 2.5, καθώς και το κρυπτογραφικό αποτύπωμα του προηγούμενου μπλοκ, δημιουργώντας έτσι μία αλυσίδα από μπλοκ. Κάθε κόμβος στο δίκτυο του blockchain έχει στην κατοχή του ένα αντίγραφο του blockchain.[27]



Εικόνα 2.1 Οπτική αναπαράσταση της σύνδεσης μεταξύ των μπλοκ.[6]

Για την ασφάλεια του blockchain χρησιμοποιούνται διάφορα πρωτόκολλα συναίνεσης όπως το Proof of work και το Proof of stake και ανάλογα με το πρωτόκολλο που χρησιμοποιείται, δημιουργούνται και προστίθενται με διαφορετικές προϋποθέσεις τα μπλοκ.[27]
Υπάρχουν τρία είδη αρχιτεκτονικών για τα συστήματα τα οποία βλέπουμε στην εικόνα 2.2.



Εικόνα 2.2 Αρχιτεκτονικές συστημάτων.[15]

Το πρώτο είδος είναι τα κεντρικοποιημένα συστήματα. Αυτά τα συστήματα χρησιμοποιούν την αρχιτεκτονική client/server όπου ένας ή περισσότεροι κόμβοι είναι απευθείας συνδεδεμένοι με τον κεντρικό server.[10]

Το δεύτερο είδος είναι τα αποκεντρικοποιημένα συστήματα. Σε αυτά τα συστήματα υπάρχουν πολλοί κεντρικοί κόμβοι, χωρίς όμως οι εξαρτημένοι κόμβοι να συνδέονται μεταξύ τους, και κάθε

κεντρικός κόμβος παίρνει τις δικές του αποφάσεις, αλλά η τελική συμπεριφορά του συστήματος εξαρτάτε από το σύνολο των αποφάσεων του κάθε κόμβου.[10]

Το τρίτο και τελευταίο είδος είναι τα κατανεμημένα συστήματα όπου δεν υπάρχουν κεντρικοί κόμβοι και όλοι οι κόμβοι είναι συνδεδεμένοι μεταξύ τους. Οι συμπεριφορά του συστήματος εξαρτάτε από τις αποφάσεις όλων των κόμβων. [10]

2.1 Διαφορές μεταξύ αρχιτεκτονικών

Παρακάτω, στον πίνακα 2.1 θα δούμε τις κύριες διαφορές μεταξύ των αρχιτεκτονικών που αναφέρθηκαν στην ενότητα 2.

	Κεντρικοποιημένα συστήματα	Αποκετρικοποιημένα συστήματα	Κατανεμημένα συστήματα
Σημεία αποτυχίας και συντήρηση	Εύκολη συντήρηση λόγω ενός μόνο σημείου αποτυχίας	Περισσότερα αλλά πεπερασμένα σημεία αποτυχίας	Δυσκολία συντήρησης λόγω ότι τα σημεία αποτυχίας είναι όσα και οι χρήστες του συστήματος
Ανοχή σε σφάλματα και σταθερότητα	Αν πάθει βλάβη ο κεντρικός κόμβος όλο το σύστημα καταρρέει	Ακόμα και αν πάθει βλάβη κάποιος κόμβος υπάρχουν πολλοί ακόμα	Ακόμα και αν πάθει βλάβη κάποιος κόμβος υπάρχουν πολλοί ακόμα
Επεκτασιμότητα	Μικρή επεκτασιμότητα, λόγο του περιορισμού των χρηστών που μπορεί να αντέξει ο server	Μέτρια επεκτασιμότητα λόγο περισσότερων κεντρικών κόμβων	Άπειρη επεκτασιμότητα, ανάλογη των χρηστών του δικτύου
Ευκολία ανάπτυξης και δημιουργίας	Εύκολο στην ανάπτυξη λόγο ότι υπάρχει μόνο η ανάγκη επιλογής της δομής	Δυσκολότερο στην ανάπτυξη λόγο της ανάγκης υλοποίησης λογισμικού για την κοινή χρήση πόρων και η επικοινωνία μεταξύ των κόμβων	Δυσκολότερο στην ανάπτυξη λόγο της ανάγκης υλοποίησης λογισμικού για την κοινή χρήση πόρων και η επικοινωνία μεταξύ των κόμβων

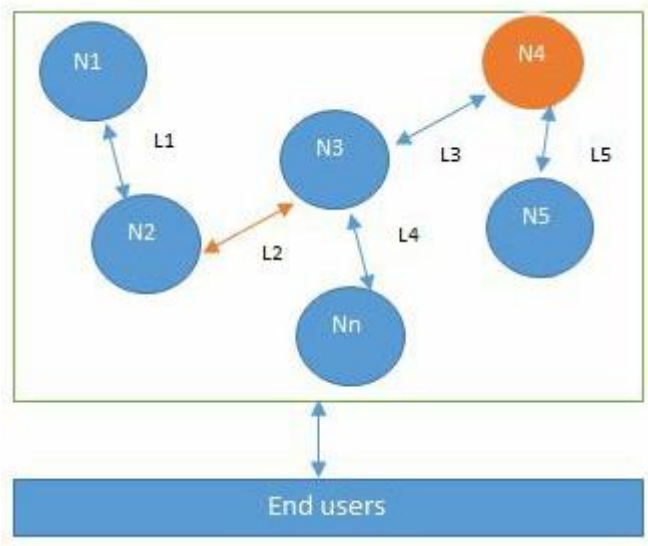
Πίνακας 2.1 Διαφορές μεταξύ κεντρικοποιημένων, αποκεντρικοποιημένων και κατανεμημένων συστημάτων[15]

2.2 Κατανεμημένα συστήματα

Είναι σημαντικό να κατανοήσει κανείς τα κατανεμημένα και τα κεντρικοποιημένα συστήματα διότι το blockchain ουσιαστικά είναι ένα αποκεντρικοποιημένο και κατανεμημένο σύστημα.[5]

Τα κατανεμημένα συστήματα είναι ένα υπολογιστικό παράδειγμα όπου δύο οι περισσότεροι κόμβοι συνεργάζονται με συντονισμένος τρόπο έτσι ώστε να πετύχουν ένα κοινό αποτέλεσμα και είναι μοντελοποιημένα έτσι ώστε οι χρήστες να τα βλέπου ως ένα σύστημα.[5]

Ένας κόμβος μπορεί να χαρακτηριστεί ως ένας μεμονωμένος χρήστης σε ένα κατανεμημένο σύστημα. Όλοι οι κόμβοι μπορούν να στέλνουν και να δέχονται μηνύματα μεταξύ του. Οι κόμβοι μπορούν να είναι έμπιστοι, ελαττωματικοί ή κακόβουλοι και έχουν ο καθένας δικιά τους μνήμη και επεξεργαστή. Ένας κόμβος ο οποίος επιδεικνύει αυθαίρετη ή απροσδόκητη συμπεριφορά ονομάζεται βυζαντινός κόμβος. Αυτή η συμπεριφορά μπορεί να είναι επιτηδευμένα κακόβουλη, η οποία είναι επιβλαβής για την λειτουργία του δικτύου. Στην εικόνα 2.3 μπορούμε να ένα παράδειγμα κατανεμημένου συστήματος με ελαττωματικά στοιχεία. Μπορούμε να διακρίνουμε τον κόμβο N4, που είναι ένας βυζαντινός κόμβος, καθώς και το σύνδεσμο L2 οποίος είναι ένας σπασμένος ή αργός σύνδεσμος.[5]



Εικόνα 2.3 Κατανεμημένο συστήματος με ελαττωματικά στοιχεία.[5]

Από αυτό μπορούμε να καταλάβουμε εύκολα τις προκλήσεις που προκύπτουν σε ένα κατανεμημένο σύστημα, το οποίο θέλουμε να λειτουργεί χωρίς ατέλειες. Τα κατανεμημένα συστήματα είναι δύσκολα στον σχεδιασμό, το οποίο έχει αποδειχθεί από το θεώρημα CAP, το οποίο λέει με λίγα λόγια ότι ένα κατανεμημένο σύστημα δεν μπορεί να έχει όλες τις επιθυμητές ιδιότητες ταυτόχρονα.[5]

2.3 Η ιστορία του blockchain

Η αρχική ιδέα που περιγράφει το blockchain είχε γίνει το 1982 από τον κρυπτογράφο David Chaum ο οποίος πρότεινε ένα πρωτόκολλο παρόμοιο με το Blockchain στην διατριβή του με τίτλο "Computer Systems Established, Maintained, and Trusted by Mutually Suspicious Groups.". [32] Το 1991 οι ερευνητές Stuart Haber και W. Scott Stornetta εισήγαγαν μία υπολογιστικά πρακτική λύση, για την χρονοσφραγίδα ψηφιακών εγγράφων, ώστε να μην μπορεί κάποιος να αλλοιώσει τις ημερομηνίες των εγγράφων. Ένα χρόνο αργότερα, το 1992 ενσωματώθηκε στο σχέδιο το Merkle tree όπου είναι δυνατή η συλλογή πολλών εγγράφων σε ένα μπλοκ, κάνοντας το πιο αποδοτικό. Παρ' όλα αυτά αυτή η τεχνολογία δεν πολυχρησιμοποιούνταν και σταμάτησε το 2004. Το 2008 παρουσιάστηκε η πρώτη εφαρμογή του blockchain μαζί με την εφεύρεση του bitcoin από Τον/τους Satoshi Nakamoto[29], ο οποίος βελτίωσε σημαντικά το Blockchain με την χρήση αλγόριθμου hashcash για την χρονοσφραγίδα των μπλοκ χωρίς να χρειάζεται η «υπογραφή» από κάποιον αξιόπιστο και πρόσθεσε επίσης μία παράμετρο δυσκολίας για να σταθεροποιηθεί ο ρυθμός που προστίθενται μπλοκ στην αλυσίδα.[22]

2.3.1 Ιστορία του Ethereum

Μετά την εφεύρεση του Bitcoin, το 2013, ένας νεαρός προγραμματιστής ο Vitalik Buterin ξεκίνησε να σκέφτεται την περεταίρω επέκταση των δυνατοτήτων του Bitcoin και του Mastercoin.[3]

Τον Οκτώβριο του ίδιου έτους πρότεινε στην ομάδα του Mastercoin μία γενικευμένη προσέγγιση που επιτρέπει πιο ευέλικτα συμβόλαια (αλλά όχι Turing πλήρες) για να αντικατασταθεί η εξειδικευμένη γλώσσα προγραμματισμού του Mastercoin για έξυπνα συμβόλαια. Παρόλο που η ιδέα εντυπωσίασε την ομάδα του Mastercoin, τελικά απορρίφθηκε λόγω ότι θα χρειαζόταν μια ριζική αλλαγή και δεν ταίριαζε στον χάρτη πορείας της ομάδας.[3]

Τον Δεκέμβριο του ίδιου έτους, ο Vitalik μοιράζεται ένα whitepaper το οποίο περιέγραφε την ιδέα του Ethereum, ένα Turing πλήρες και γενικής χρήσης Blockchain. Στην συνέχεια ο Vitalik μαζί με τον Gavin Wood άρχισαν να εξελίζουν την ιδέα και ανέπτυξαν μαζί την επίπεδο πρωτοκόλλου που στην συνέχεια έγινε το Ethereum.[3]

Οι δύο ιδρυτές δούλεψαν πάνω σ ’αυτό το project για δύο χρόνια και στης 30 Ιουλίου του 2015 εξορύχθηκε το πρώτο μπλοκ και ο παγκόσμιος υπολογιστής του Ethereum ξεκίνησε να εξυπηρετεί τον κόσμο.[3]

2.4 Κλειδιά και διευθύνσεις

Η ιδιοκτησία των κρυπτονομισμάτων, όπως του ether, καθιερώνεται μέσω των ψηφιακών ιδιωτικών κλειδιών, των διευθύνσεων και των ψηφιακών υπογραφών. Τα ιδιωτικά κλειδιά όμως, είναι η καρδία όλων των αλληλεπιδράσεων των χρηστών. Πιο συγκεκριμένα οι διευθύνσεις των λογαριασμών παράγονται από τα ιδιωτικά κλειδιά, κάθε ιδιωτικό κλειδί αντιστοιχεί σε μία μοναδική διεύθυνση ή αλλιώς σε έναν λογαριασμό.[3]

Τα ιδιωτικά κλειδιά πρέπει να είναι πάντα κρυφά και να μην εμφανίζονται σε μηνύματα που στέλνονται στο δίκτυο και ούτε να αποθηκεύονται στο blockchain. Αυτά που μπορούν να μεταδοθούν και να αποθηκεύονται είναι οι διευθύνσεις των λογαριασμών και οι ψηφιακές υπογραφές.[3]

Η πρόσβαση και ο έλεγχος στα κρυπτονομίσματα επιτυγχάνεται με τις ψηφιακές υπογραφές, κάθε συναλλαγή χρειάζεται μία έγκυρη ψηφιακή υπογραφή για να συμπεριληφθεί στο blockchain. Οποιοσδήποτε που έχει ένα αντίγραφο του ιδιωτικού κλειδιού έχει τον έλεγχο του λογαριασμού και τα κρυπτονομίσματα που υπάρχουν σε αυτόν. Οι ψηφιακές υπογραφές στο blockchain, αποδεικνύουν τον ιδιοκτήτη των νομισμάτων, επειδή αποδεικνύουν την ιδιοκτησία του ιδιωτικού κλειδιού.[3]

Σε συστήματα που χρησιμοποιούν κρυπτογραφία δημόσιου κλειδιού, όπως το blockchain, τα κλειδιά έρχονται σε ζευγάρια που αποτελούνται από το ιδιωτικό κλειδί και το δημόσιο κλειδί. Σαν παρομοίωση στην καθημερινή ζωή θα μπορούσε να πει κανείς ότι το δημόσιο κλειδί είναι το IBAN και το ιδιωτικό το PIN που έχουμε στην κάρτα μας, το πρώτο χρησιμοποιείται για την αναγνώριση ενός λογαριασμού από τους άλλους και το δεύτερο για τον έλεγχο των χρημάτων του λογαριασμού.[3]

Όσον αφορά το κομμάτι των πληρωμών στις συναλλαγές του blockchain, ο επιθυμητός παραλήπτης αντιπροσωπεύεται από την διεύθυνση του, και λειτουργεί όπως τα προσωπικά στοιχεία του λογαριασμού του δικαιούχου σε μία τραπεζική συναλλαγή. Στο blockchain του Ethereum, η διεύθυνση ενός χρήστη παράγεται από το δημόσιο κομμάτι του ζευγαριού, όμως στο Ethereum

υπάρχουν διευθύνσεις που δεν αντιπροσωπεύονται από ζευγάρια δημόσιων-ιδιωτικών κλειδιών, οι διευθύνσεις αυτές είναι των έξυπνων συμβολαίων, τα οποία έχουν μόνο δημόσια κλειδιά.[3]

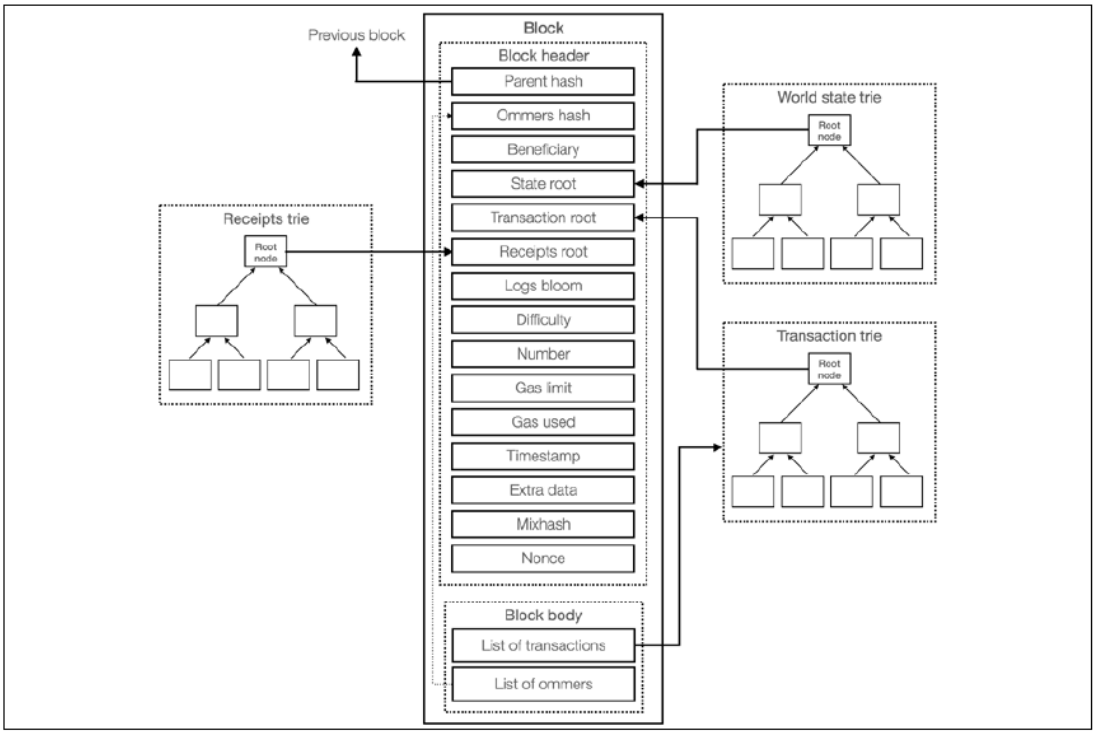
2.5 Τα μπλοκ της αλυσίδας

Τα μπλοκ του Blockchain είναι μία δομή δεδομένων στην οποία περιέχονται οι συναλλαγές που γίνονται στο Blockchain. Κάθε μπλοκ αποτελείτε από την επικεφαλίδα της, η οποία περιλαμβάνει διάφορα metadata καθώς και οι συναλλαγές που καταγράφτηκαν στο μπλοκ.[3]

Η δομή ενός μπλοκ του Ethereum έχει ως εξής :

- Την επικεφαλίδα του μπλοκ
- Την λίστα με τις συναλλαγές
- Την λίστα με τα ommer μπλοκ (ή αλλίως θείους).

Στην εικόνα 2.4 μπορούμε να δούμε την απεικόνιση της δομής ενός μπλοκ.



Εικόνα 2.4 Δομή ενός μπλοκ[6]

2.5.1 Η επικεφαλίδα του μπλοκ

Η επικεφαλίδα ενός μπλοκ αποτελείτε από διάφορά στοιχεία τα οποία θα δούμε παρακάτω.

- Το κρυπτογραφικό αποτύπωμα του γονικού μπλοκ (δηλαδή του προηγούμενου μπλοκ στην αλυσίδα).
- Το κρυπτογραφικό αποτύπωμα των ommer μπλοκ, τα οποία είναι συγγενικά μπλοκ από δευτερεύοντες αλυσίδες.
- Την διεύθυνση του δικαιούχου που θα λάβει την ανταμοιβή από το επιτυχώς εξορυγμένο μπλοκ.
- Το κρυπτογραφικό αποτύπωμα της ρίζας του δέντρου merkle με τις καταστάσεις που υπάρχουν στο EVM και υπολογίζεται αφού όλες οι συναλλαγές έχουν επεξεργαστεί και ολοκληρωθεί.

- Το κρυπτογραφικό αποτύπωμα της ρίζας του δέντρου merkle με όλες τις συναλλαγές που έχουν εισαχθεί στο μπλοκ.
- Το κρυπτογραφικό αποτύπωμα της ρίζας του δέντρου merkle με τις αποδείξεις των συναλλαγών που έχουν εισαχθεί στο μπλοκ.
- Τις καταγραφές bloom, τα οποία είναι φίλτρα bloom και αποτελούνται από την διεύθυνση του καταγραφέα και διάφορα καταγραφών κάθε απόδειξης συναλλαγής.
- Το επίπεδο δυσκολίας του μπλοκ.
- Τον συνολικό αριθμό των προηγούμενων μπλοκ.
- Το όριο του gas που έχει τεθεί για κάθε μπλοκ.
- Το gas που χρησιμοποιήθηκε από τις συναλλαγές που εμπεριέχονται στο μπλοκ.
- Την χρονοσφραγίδα, δηλαδή την χρονική στιγμή που δημιουργήθηκε το μπλοκ.
- Έξτρα δεδομένα, τα οποία είναι αυθαίρετα δεδομένα σχετικά με το μπλοκ.
- Το mixhash, το οποίο περιλαμβάνει τον 256 bit κατακερματισμό, ο οποίος όταν συνδυάζεται με τον αριθμό nonce αποδεικνύει ότι έχει δαπανηθεί υπολογιστική ισχύς μέσο του PoW (Proof of Work) αλγόριθμού για την δημιουργία του μπλοκ.
- Τον αριθμό nonce, ο οποίος είναι ένας αριθμός που σε συνδυασμό με το mixhash αποδεικνύει ότι έχει δαπανηθεί υπολογιστική ισχύς μέσο του PoW (Proof of Work) αλγόριθμού για την δημιουργία του μπλοκ.[6]

2.5.2 Τα αναγνωστικά ενός μπλοκ

Για να ξεχωρίσουμε ένα μπλοκ υπάρχουν δύο τρόποι. Ο πρώτος τρόπος είναι το κύριο χαρακτηριστικό ενός μπλοκ, το κρυπτογραφικό αποτύπωμα του, το οποίο φτιάχνεται κατακερματίζοντας(hashing) την επικεφαλίδα του μπλοκ δύο φορές χρησιμοποιώντας τον αλγόριθμο KECCAK256. Αυτό έχει σαν αποτέλεσμα ένα κρυπτογραφικό αποτύπωμα μεγέθους 32 byte το οποίο ονομάζεται κρυπτογραφικό αποτύπωμα του μπλοκ, το οποίο είναι μοναδικό για κάθε μπλοκ και μπορεί να παραχθεί από οποιοδήποτε κόμβο.[2]

Αξίζει να αναφερθεί ότι το κρυπτογραφικό αποτύπωμα δεν εμπεριέχεται στην δομή δεδομένων του μπλοκ, δεν μεταδίδεται μαζί με το μπλοκ στο δίκτυο, αλλά ούτε αποθηκεύεται στον αποθηκευτικό χώρο ενός κόμβου ως μέρος του blockchain. Αντί αυτού το κρυπτογραφικό αποτύπωμα ενός μπορεί να αποθηκευτεί σε μία ξεχωριστή βάση δεδομένων σαν μέρος των μεταδεδομένων του block για την ταχύτερη ανάκτηση των μπλοκ από τον δίσκο.[2]

Ο δεύτερος τρόπος είναι από το ύψος του μπλοκ, η αλλιώς την θέση του στο Blockchain. Προφανώς σε ύψος μηδέν θα είναι το πρώτο μπλοκ που δημιουργήθηκε και κάθε μπλοκ που δημιουργείτε είναι μία θέση παραπάνω.[2]

Ενώ το κρυπτογραφικό αποτύπωμα είναι μοναδικό για κάθε μπλοκ, το ύψος του μπλοκ δεν είναι καθώς δύο η περισσότερα μπλοκ μπορούν να έχουν το ίδιο ύψος και να ανταγωνίζονται μεταξύ τους για την ίδια θέση στο blockchain, λόγο των διακλαδώσεων του blockchain(blockchain forks). Το ύψος ενός μπλοκ, όπως και το κρυπτογραφικό του αποτύπωμα, δεν περιέχεται στην δομή δεδομένων του μπλοκ, κάθε κόμβος αναγνωρίζει δυναμικά το ύψος του μπλοκ όταν το λαμβάνει από το δίκτυο. Το ύψος του μπλοκ μπορεί επίσης να αποθηκευτεί σαν μεταδεδομένο σε μία βάση δεδομένων για ταχύτερη ανάκτηση. [2]

2.5.3 Το μπλοκ γένεσης(Genesis block)

Το πρώτο μπλοκ στο blockchain ονομάζεται το μπλοκ γένεσης και είναι ο κοινός απόγονος όλων των μπλοκ, πράγμα που σημαίνει ότι άμα ακολουθήσουμε την αλυσίδα προς τα πίσω κάποια στιγμή θα φτάσουμε σε αυτό. Είναι το μοναδικό μπλοκ το οποίο δεν κάνει αναφορά στο προηγούμενο του, διότι δεν υπάρχει. Κάθε κόμβος έχει τουλάχιστον ένα μπλοκ επειδή το μπλοκ της γένεσης είναι κωδικοποιημένο στον client του λογισμικού του blockchain και δεν μπορεί να τροποποιηθεί, κάθε κόμβος γνωρίζει το κρυπτογραφικό αποτύπωμα, την δομή αυτού του μπλοκ, την χρονική στιγμή που δημιουργήθηκε καθώς και την μοναδική συναλλαγή σε αυτό, έτσι κάθε κόμβος έχει μία κοινή αφετηρία από την οποία χτίζεται ένα blockchain εμπιστοσύνης.[2]

2.5.4 Τα δέντρα του merkle

Κάθε μπλοκ περιέχει συνοπτικά όλες τις συναλλαγές και τις αποδείξεις που έγιναν μέχρι την δημιουργία του μπλοκ και όλες τις καταστάσεις EVM (όπως υπόλοιπα λογαριασμών, κώδικα συμβολαίων και άλλα) χρησιμοποιώντας ένα δέντρο του merkle.

Ένα δέντρο merkle είναι μία δομή δεδομένων για την αποδοτική σύνοψη και την επαλήθευση της ακεραιότητας μεγάλων ομάδων από δεδομένα. Στο blockchain, τα δέντρα merkle είναι δυαδικά δέντρα τα οποία περιέχουν κρυπτογραφικά αποτυπώματα από συναλλαγές. Ο όρος «δέντρο» χρησιμοποιείτε στην επιστήμη των υπολογιστών για να περιγράψουν μία δομή δεδομένων η οποία έχει διακλαδώσεις, συνήθως όμως αυτά τα δέντρα παρουσιάζονται ανάποδα, με την ρίζα στην κορυφή και τα «φύλλα» στο κάτω μέρος του διαγράμματος.[2]

Ένα δέντρο merkle κατασκευάζεται αναδρομικά εφαρμόζοντας δύο φορές τον αλγόριθμο KECCAK256 σε ζευγάρια από κόμβους μέχρι να μείνει μόνο ένας, ο οποίος ονομάζεται ρίζα του merkle.[2]

Για N αριθμό δεδομένων σε ένα δέντρο merkle, χρειάζονται το πολύ 2*log2(N) υπολογισμοί για να ελέγξουμε αν ένα συγκεκριμένο δεδομένο περιέχεται στο δέντρο, κάνοντας τα δέντρα merkle μία πάρα πολύ αποδοτική δομή δεδομένων.[2]

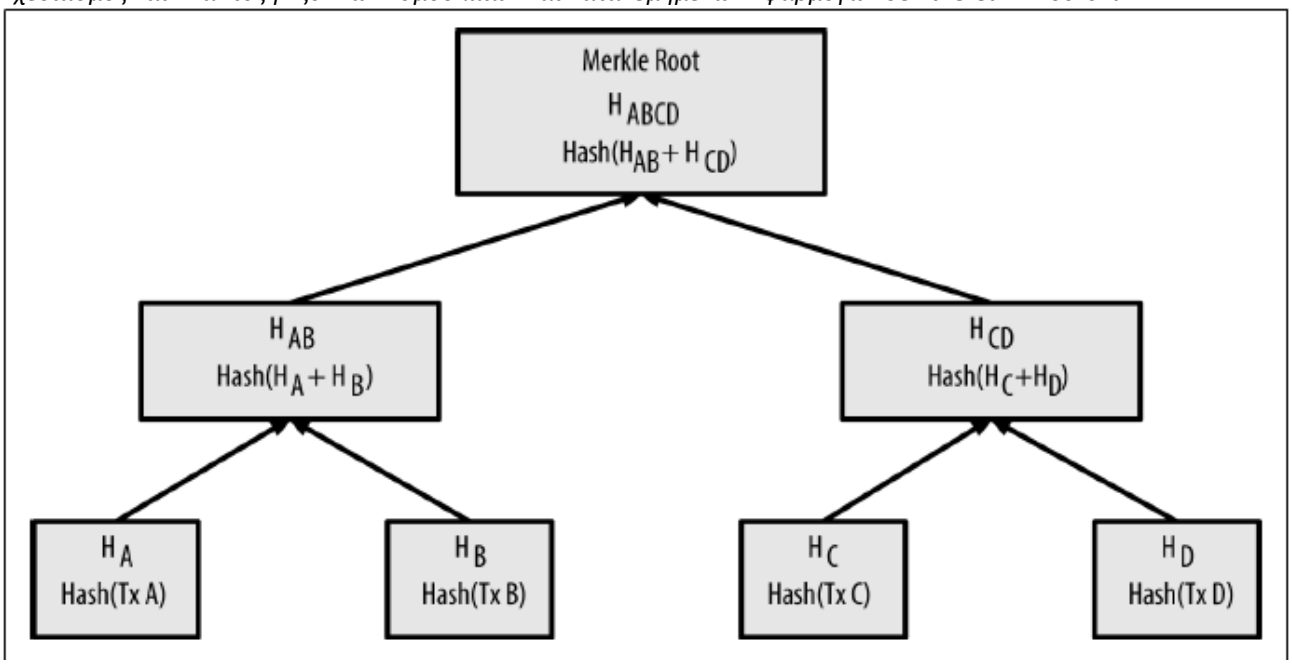
Για παράδειγμα, ας υποθέσουμε πώς έχουμε τέσσερεις συναλλαγές την A, B, C και D, οι οποίες αποτελούν τα φύλλα ενός δέντρου merkle, αφότου έχουν κατακερματιστεί πρώτα, δύο φορές, με τον αλγόριθμο KECCAK256, με αποτέλεσμα τέσσερα φύλλα H_A, H_B, H_C και H_D.[2]

Όπου για παράδειγμα H_A = KECCAK256 (KECCAK256 (συναλλαγή A))

Συνεχίζοντας, συνενώνουμε ζευγάρια από κατακερματισμένες συναλλαγές και τις κατακερματίζουμε, δύο φορές, μαζί για να φτιάξουμε έναν μητρικό κόμβο.

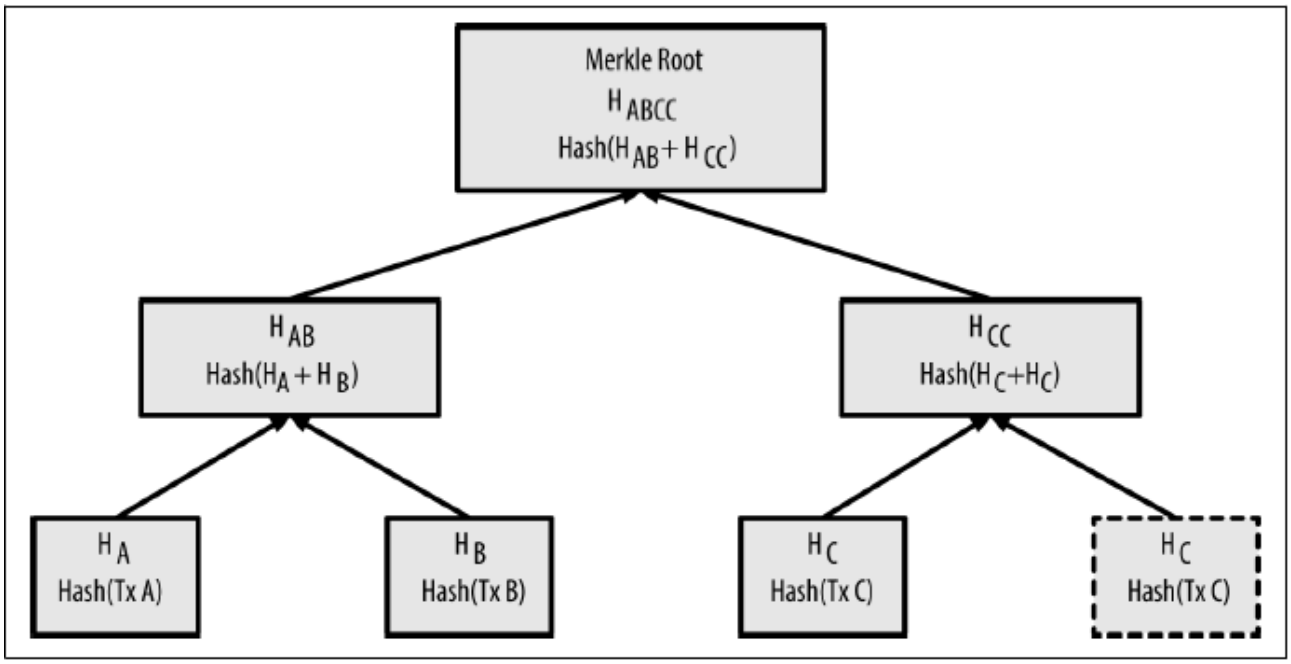
Για παράδειγμα H_{AB} = KECCAK256 (KECCAK256 (H_A + H_B))[2]

Όπως είπαμε, αυτή η διαδικασία συνεχίζεται μέχρι να μείνει μόνο ένας κόμβος, του οποίου το κρυπτογραφικό αποτύπωμα αποθηκεύεται στην επικεφαλίδα του μπλοκ και συνοψίζει τα δεδομένα και των τεσσάρων συναλλαγών όπως μπορούμε να δούμε στην εικόνα 2.5.[2]



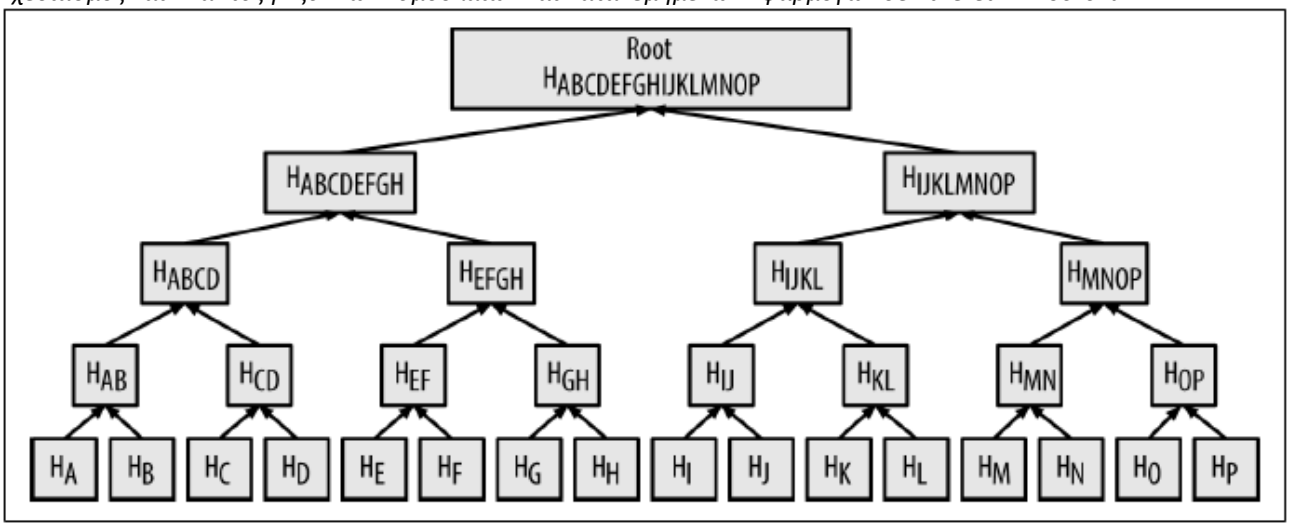
Εικόνα 2.5 Διάγραμμα δέντρου merkle.[2]

Τα δέντρα merkle είναι δυαδικά, γι' αυτό χρειάζονται ένα ζυγό αριθμό από φύλλα. Αν ο αριθμός των συναλλαγών είναι μόνος, τότε θα γίνει ένα αντίγραφο της τελευταίας συναλλαγής, έτσι ώστε ο αριθμός των συναλλαγών να γίνει ζυγός. Ένα δέντρο το οποίο έχει ζυγό αριθμό από φύλλα ονομάζεται ισορροπημένο (εικόνα 2.6).[2]



Εικόνα 2.6 Αντιγραφή της τελευταίας συναλλαγής για την επίτευξη ισορροπημένου δέντρου[2]

Αυτή η μέθοδος μπορεί να χρησιμοποιηθεί για την κατασκευή δέντρων οποιοδήποτε μεγέθους με το ίδιο τρόπο παράγοντας 32 bytes δεδομένων ως μία ρίζα merkle. Παρακάτω στην εικόνα 2.7 θα δούμε ένα δέντρο merkle με δεκαέξι συναλλαγές. Παρόλο που κάποιος θα υποψιαζόταν ότι η ρίζα έχει μεγαλύτερο όγκο δεδομένων διότι περιέχει περισσότερες συναλλαγές, αυτό που ισχύει είναι πώς έχει το ίδιο μέγεθος με ένα δέντρο με τέσσερα φύλλα, δηλαδή 32 bytes. Αυτό συμβαίνει επειδή όσες συναλλαγές και αν περιέχονται σε ένα δέντρο merkle η ρίζα του πάντα συνοψίζει τα δεδομένα σε 32 bytes.[2]



Εικόνα 2.7 Δέντρο merkle με 16 φύλλα[2]

Για να αποδειχθεί αν μία συγκεκριμένη συναλλαγή εμπεριέχεται σε ένα μπλοκ, ένας κόμβος θα παράξει log2(N) κατακερματισμούς των 32 bytes, τα οποία αποτελούν την διαδρομή merkle και συνδέουν την συγκεκριμένη συναλλαγή με την ρίζα του δέντρου. Αυτό είναι ιδιαίτερα σημαντικό καθώς αυξάνουν οι συναλλαγές, διότι ο λογάριθμος με βάση το δύο του αριθμού των συναλλαγών αυξάνεται πολύ αργά.[2]

Στον πίνακα 2.2 μπορούμε να δούμε πόσο γρήγορα μεταβάλλεται το μέγεθος του block και το μέγεθος της διαδρομής σε αριθμό υπολογισμών κατακερματισμού και bytes ανάλογα με τον αριθμό των συναλλαγών.[2]

Number of transactions	Approx. size of block	Path size (hashes)	Path size (bytes)
16 transactions	4 kilobytes	4 hashes	128 bytes
512 transactions	128 kilobytes	9 hashes	288 bytes
2048 transactions	512 kilobytes	11 hashes	352 bytes
65,535 transactions	16 megabytes	16 hashes	512 bytes

Πίνακας 2.2 αποδοτικότητα δέντρου merkle.[2]

Μπορούμε εύκολα να παρατηρήσουμε ότι ενώ το μέγεθος του block αυξάνεται πολύ γρήγορα σε σύγκριση με το μέγεθος της διαδρομής σε hash και σε bytes, τα οποία αντιθέτως αυξάνονται πολύ αργά.[2]

2.6 Mining και συναίνεση

Mining είναι η διαδικασία με την οποία προστίθενται μπλοκ στο blockchain, μεγαλώνοντας την αλυσίδα συνεχώς, Από ειδικούς κόμβους οι οποίοι ονομάζονται miners, οι οποίοι επικυρώνουν συναλλαγές, δημιουργούν υποψήφια μπλοκ και προσπαθούν να τα επαληθεύσουν με την χρήση του αλγόριθμου PoW, ο οποίος είναι αρκετά απαιτητικός σε πόρους συστήματος. Το αποτέλεσμα, να προσφέρουν ασφάλεια στο σύστημα από απάτες και double spending, ενώ παράλληλα προσθέτουν περισσότερα νομίσματα στο οικοσύστημα του blockchain.[5]

Η συναίνεση του δικτύου blockchain επιτυγχάνεται μέσο της ανεξάρτητης επαλήθευσης των συναλλαγών, από κάθε κόμβο, ελέγχοντας αν πληρούν κάποια συγκεκριμένα κριτήρια, από την ανεξάρτητη καταγραφή συναλλαγών σε υποψήφια μπλοκ από κόμβους mining σε συνδυασμό με αποδεδειγμένη δαπανημένη υπολογιστική προσπάθεια, μέσο του αλγόριθμου PoW, ανεξάρτητη

επαλήθευση των υποψήφιων μπλοκ από κάθε κόμβο, η σύνδεση του μπλοκ στην αλυσίδα και η ανεξάρτητη επιλογή, από κάθε κόμβο, της διακλάδωσης της αλυσίδας με την μεγαλύτερη επαληθευμένη δαπάνη υπολογιστικής προσπάθειας μέσο του αλγόριθμου PoW.[2]

2.6.1 Mining κόμβοι

Όπως αναφέραμε, οι mining κόμβοι είναι ειδικοί κόμβοι, οι οποίοι εκτελούν την διαδικασία του mining.

Τα καθήκοντα τους είναι τα εξής :

- Να συγχρονίζονται με το δίκτυο. Όταν ένας νέος κόμβος συνδέεται στο δίκτυο, κατεβάζει το blockchain ζητώντας τα μπλοκ από τους υπόλοιπους κόμβους.

- Η επικύρωση συναλλαγών οι οποίες αναμεταδίδονται στο δίκτυο. Η επικύρωση των συναλλαγών γίνεται μέσω της επαλήθευσης και επικύρωσης των υπογραφών και των εξόδων, των συναλλαγών.

- Η επικύρωση των μπλοκ, η οποία γίνεται ελέγχοντας αν ακολουθούν κάποιους συγκεκριμένους κανόνες, καθώς επίσης και η επαλήθευση κάθε συναλλαγής στο μπλοκ και η επαλήθευση της τιμής nonce.

- Η δημιουργία ενός νέου μπλοκ, στην οποία οι miners προτείνουν έναν νέο μπλοκ το οποίο περιλαμβάνει τις συναλλαγές που έχουν αναμεταδοθεί στο δίκτυο, αφού πρώτα τις έχουν επαληθεύσει.

- Εκτέλεση του PoW, το οποίο είναι κυριότερο καθήκον της διεργασίας του mining, στο οποίο οι miners βρίσκουν έγκυρα μπλοκ μέσο της επίλυσης υπολογιστικών παζλ. Η επικεφαλίδα του block περιέχει το nonce, όπως είχαμε αναφέρει σε προηγούμενο κεφάλαιο, με 32 bit, το οποίο οι miners χρειάζεται να μεταβάλουν συνέχεια την τιμή του μέχρι ο κατακερματισμός να λάβει τιμή μικρότερη από έναν προκαθορισμένο στόχο, ο οποίος εξαρτάται από τον στόχο δυσκολίας του block.

- Και τελευταία, να λάβουν την ανταμοιβή τους. Μόλις ένας κόμβος επιλύσει το παζλ του κατακερματισμού, αναμεταδίδει άμεσα τα αποτελέσματα και αφού οι υπόλοιποι κόμβοι επαληθεύσουν, αποδέχονται το καινούργιο μπλοκ το οποίο προστίθεται στο blockchain. Υπάρχει μία μικρή πιθανότητα το νέο μπλοκ να μην γίνει αποδεκτό άμεσα λόγο ότι μπορεί περίπου την ίδια στιγμή να ανακαλυφθεί ένα άλλο μπλοκ. Στην περίπτωση που ένα μπλοκ γίνει αποδεκτό τότε ο miner λαμβάνει την ανταμοιβή του.[5]

2.6.2 Δημιουργία του μπλοκ

Ένας miner, όπως και κάθε κόμβος, συλλέγει, επικυρώνει και αναμεταδίδει νέες συναλλαγές, όμως εκτός αυτών καταγράφει τις συναλλαγές σε ένα υποψήφιο μπλοκ, το οποίο γίνεται αποδεκτό από τους υπόλοιπους κόμβους μόνο εάν ο miner βρει την λύση στο PoW αλγόριθμο.[2]

Το υποψήφιο μπλοκ, περιέχει όλες τις συναλλαγές που έχουν γίνει απ’ την στιγμή που εξορύχθηκε το τελευταίο μπλοκ στην αλυσίδα μέχρι να φτάσει το υποψήφιο μπλοκ μία προκαθορισμένη τιμή gas (στην περίπτωση του Ethereum). Όλες οι επικυρωμένες συναλλαγές τοποθετούνται από τον κόμβο σε μία ομάδα συναλλαγών, από την οποία καταγράφει τις συναλλαγές στο υποψήφιο μπλοκ. Μόλις το μέγεθος (ή το gas) τού μπλοκ φτάσει στο όριο του, δεν καταγράφονται οι συναλλαγές που απόμειναν και περιμένουν για να καταγραφούν στο επόμενο.[2]

Κατά την δημιουργία ενός υποψήφιου μπλοκ δημιουργείται και μία ειδική συναλλαγή, η οποία ονομάζεται coinbase, είναι η πρώτη συναλλαγή του μπλοκ και περιέχει την ανταμοιβή του miner. Αυτή η συναλλαγή δεν έχει εισόδους που καταναλώνουν νομίσματα, αντί αυτών, έχει μία είσοδο, η οποία ονομάζεται coinbase, η οποία δημιουργεί κρυπτονομίσματα από το τίποτα και έχει μία έξοδο, την διεύθυνση του miner.[2]

2.6.3 Εξόρυξη και Proof of Work

Μόλις δημιουργηθεί το υποψήφιο μπλοκ, ο miner προσπαθεί να το «εξορύξει», βρίσκοντας την λύση στον αλγόριθμο PoW, η οποία κάνει το μπλοκ έγκυρο.[2]

Με λίγα λόγια η εξόρυξη είναι η διαδικασία όπου η επικεφαλίδα του μπλοκ κατακερματίζεται μέχρι το αποτέλεσμα να είναι μικρότερο από έναν συγκεκριμένο στόχο, ο οποίος αλλάζει ανάλογα από τον στόχο δυσκολίας. Το αποτέλεσμα της συνάρτησης κατακερματισμού δεν έχει κάποιο μοτίβο, δεν μπορεί να δημιουργηθεί κάποιο μοτίβο το οποίο δημιουργεί μία συγκεκριμένη τιμή κατακερματισμού και δεν μπορεί να προβλεφθεί. Ο μόνος τρόπος είναι η επαναλαμβανόμενη προσπάθεια επίλυσης του μεταβάλλοντας την τιμή nonce, μέχρι να παραχθεί τυχαία η επιθυμητή τιμή κατακερματισμού.[2]

Ο αλγόριθμος PoW είναι ένας αλγόριθμος κατακερματισμού ο οποίος παίρνει σαν είσοδο δεδομένα αυθαίρετου μήκους και παράγει ένα ντετερμινιστικό αποτέλεσμα. Για κάποια συγκεκριμένη είσοδο το αποτέλεσμά του θα είναι πάντα το ίδιο και είναι εύκολα υπολογίσιμο και επαληθεύσιμο από οποιοδήποτε εφαρμόσει τον ίδιο αλγόριθμο κατακερματισμού. Το βασικότερο χαρακτηριστικό αυτού του κρυπτογραφικού αλγόριθμου κατακερματισμού είναι ότι είναι υπολογιστικά αδύνατο να βρεθούν δύο διαφορετικές είσοδοι που να παράγουν το ίδιο αποτέλεσμα όπως επίσης είναι αδύνατο να βρούμε την είσοδο που θα μας έδινε μία επιθυμητή έξοδο.[2]

Ο στόχος δυσκολίας είναι μία τιμή, όπου για να θεωρηθεί έγκυρο ένα μπλοκ πρέπει η επικεφαλίδα του να έχει τιμή μικρότερη από αυτή. Όσο μικραίνουμε τον στόχο, η δυσκολία για την εύρεση κατακερματισμού μικρότερου από αυτόν, αυξάνεται και επηρεάζει το χρόνο που χρειάζεται για να βρεθεί λύση στον αλγόριθμο Proof of Work. Αυτός ο χρόνος είναι και η συχνότητα που εκδίδονται νέα νομίσματα και η ταχύτητα που εκτελούνται πλήρως οι συναλλαγές. Ο λόγος που μεταβάλλεται, είναι λόγο της αύξησης της ισχύς τον υπολογιστών, καθώς επίσης λόγο ότι ο αριθμός των mining κόμβων συνεχώς αλλάζει. Η μεταβολή του στόχου δυσκολίας, ο οποίος μπορεί είτε να αυξηθεί είτε να μειωθεί, γίνεται αυτόματα σε κάθε κόμβο κάθε φορά που ανακαλύπτονται 2016 μπλοκ, για να προσαρμοστεί έτσι ώστε ένα νέο μπλοκ να ανακαλύπτεται κάθε 15 δευτερόλεπτα κατά μέσο όρο.

Μέσο του αλγόριθμου Proof of Work, μπορεί ένας miner να αποδείξει ότι δαπάνησε υπολογιστική ισχύ για τον υπολογισμό της τιμής nonce, η οποία επιλύει το υπολογιστικό παζλ.[2]

2.6.4 Επιτυχημένη εξόρυξη και επικύρωση μπλοκ

Μόλις κάποιος miner βρει λύση στο υπολογιστικό παζλ, μεταδίδει την λύση που βρήκε στους υπόλοιπους κόμβους, οι οποίοι με την σειρά τους λαμβάνουν, επαληθεύουν και διαδίδουν το νέο μπλοκ. Κατά την διάδοση του μπλοκ, κάθε κόμβος το προσθέτει στο δικό αντίγραφό του blockchain, αυξάνοντας το μέγεθος της αλυσίδας. Όταν ένας miner λάβει και επικυρώσει το μπλοκ, εγκαταλείπει την προσπάθεια εύρεσης λύσης για το υποψήφιο μπλοκ που βρίσκεται στο ίδιο ύψος και ξεκινάει άμεσα τους υπολογισμούς για το επόμενο μπλοκ, χρησιμοποιώντας το νέο μπλοκ σαν γονικό. Όταν οι υπόλοιποι miners συνεχίζουν να προσθέτουν μπλοκ στην διακλάδωση της αλυσίδας από αυτό το μπλοκ, ουσιαστικά ψηφίζουν με την υπολογιστική τους ισχύ και υποστηρίζουν ότι αυτό το μπλοκ και αυτή η διακλάδωση είναι η σωστή.[2]

Η επικύρωση του μπλοκ γίνεται με τον έλεγχο κάποιων συγκεκριμένων κριτηρίων. Αυτά τα κριτήρια εξασφαλίζουν ότι μόνο έγκυρα μπλοκ θα διαδοθούν στο δίκτυο του blockchain, ότι τα μπλοκ των τίμιων miner θα προστεθούν στο blockchain με αποτέλεσμα να λάβουν την ανταμοιβή τους και ότι τα μπλοκ των δόλιων miner θα απορριφθούν, οι οποίοι εκτός από την ανταμοιβή, σπατάλησαν επίσης υπολογιστική ισχύ για την εύρεση λύσης για τον PoW, με αποτέλεσμα κόστος για το σπαταλημένο ρεύμα χωρίς αποζημίωση.[2]

2.6.5 Επιλέγοντας αλυσίδα από μπλοκ

Οι κόμβοι διατηρούν τρείς ομάδες από μπλοκ, αυτά που είναι συνδεδεμένα στην κύρια αλυσίδα του blockchain, τα μπλοκ που σχηματίζουν διακλαδώσεις (δευτερεύοντες αλυσίδες) και τα μπλοκ που δεν έχουν γνωστό γονικό μπλοκ (ορφανά). Τα μπλοκ τα οποία απορρίπτονται λόγο ότι δεν πληρούν κάποιο/α κριτήριο δεν προστίθενται σε καμία από αυτές τις ομάδες.[2]

Η κύρια αλυσίδα, είναι η αλυσίδα, η οποία καταρχάς είναι έγκυρη, και έχει την πιο δαπανημένη ισχύ λόγο του PoW. Συνήθως η κύρια αλυσίδα είναι και αυτή με τα περισσότερα μπλοκ, εκτός αν υπάρχει κάποια αλυσίδα με ίδιο μέγεθος.[2]

Η κύρια αλυσίδα έχει επίσης παρακλάδια, τις δευτερεύοντες αλυσίδες, των οποίων τα μπλοκ είναι συγγενικά των μπλοκ της κύριας αλυσίδας. Αυτά τα μπλοκ δεν είναι μέρος της κύριας αλυσίδας αλλά φυλάγονται για μελλοντική αναφορά, σε περίπτωση που κάποια από αυτές τις αλυσίδες ξεπεράσει σε δαπανημένη ισχύ την κύρια αλυσίδα.[2]

Κάθε φορά που ανακαλύπτεται ένα μπλοκ, οι κόμβοι θα την τοποθετήσουν στην αλυσίδα βάση του κρυπτογραφικού αποτυπώματος του μητρικού μπλοκ, το οποίο συνήθως είναι στην κορυφή του blockchain, επεκτείνοντας το. Όμως υπάρχουν περιπτώσεις που το νέο μπλοκ δεν τοποθετείται στην κύρια αλυσίδα, αλλά σε κάποια δευτερεύουσα επεκτείνοντάς την. Σε περίπτωση που η δευτερεύουσα αλυσίδα συγκεντρώσει περισσότερη δαπανημένη ισχύ από την κύρια, οι δύο αλυσίδες θα αλλάξουν ρόλους, κάνοντας την δευτερεύουσα την νέα κύρια αλυσίδα.[2]

Αν ανακαλυφθεί κάποιο μπλοκ και δεν βρεθεί το γονικό του, αυτό το μπλοκ θεωρείται ορφανό και αποθηκεύεται σε μία ομάδα ορφανών μπλοκ. Μόλις ανακαλυφθεί το γονικό του μπλοκ και συνδεθεί σε κάποια από τις αλυσίδες, το ορφανό μπλοκ θα μεταφερθεί από την ομάδα και θα συνδεθεί με το γονικό του, κάνοντας το μέρος της αλυσίδας. Τα ορφανά μπλοκ συνήθως δημιουργούνται όταν δύο μπλοκ ανακαλύπτονται σε μικρό χρονικό διάστημα σε ανάποδη σειρά, το οποίο έχει σαν αποτέλεσμα να δημιουργηθεί πρώτα το παιδί και μετά το γονικό.[2]

Αυτές οι διακλαδώσεις, δημιουργούνται επειδή το blockchain είναι μία αποκεντρικοποιημένη δομή δεδομένων και τα διάφορά αντίγραφα του δεν είναι πάντα τα ίδια. Τα μπλοκ μπορεί να φτάσουν σε διαφορετικούς κόμβους σε διαφορετικές στιγμές, με αποτέλεσμα κάθε κόμβος να έχει διαφορετική οπτική γωνία του blockchain. Για να επιλυθεί αυτό, κάθε κόμβος επιλέγει και προσπαθεί να προεκτείνει την αλυσίδα με την περισσότερη δαπανημένη ισχύ. Με την επιλογή της αλυσίδας με την περισσότερη δαπανημένη ισχύ ως κύρια, όλοι οι κόμβοι, κάποια στιγμή, συνήθως μετά από ένα μπλοκ, φτάνουν σε συναίνεση και όλες οι διακλαδώσεις επιλύονται, επεκτείνοντας μία από τις αλυσίδες. Οι miners με την προσθήκη ενός μπλοκ, ουσιαστικά ψηφίζουν, με την υπολογιστική τους ισχύ την αλυσίδα που θεωρούν σωστή.[2]

2.7 Πλεονεκτήματα, μειονεκτήματα και απειλές

Όπως και κάθε τι, έτσι και το blockchain έχει τα πλεονεκτήματα του που το κάνει να ξεχωρίζει έναντι άλλων τεχνολογιών, μειονεκτήματα που δημιουργούν αμφιβολίες για την

αποτελεσματικότητα του, καθώς και απειλές από κακόβουλους χρήστες, οι οποίοι είτε θεωρητικά είναι αδύνατο να συμβούν, είτε χρειάζονται τρόποι αντιμετώπισης.

2.7.1 Τα πλεονεκτήματα

Αποκεντρικοποίηση : Το κύριο πλεονέκτημα του blockchain είναι ότι είναι ένα αποκεντρικοποιημένο και κατανεμημένο σύστημα. Γι' αυτό η βάση δεδομένων του είναι διαμοιρασμένη σε όλους τους κόμβους χωρίς την ανάγκη κεντρικού διαχειριστή. Η λογική του blockchain δεν έχει την λογική μίας κεντρικοποιημένης εφαρμογής, αν' αυτού οι συναλλαγές του blockchain έχουν την δικιά τους απόδειξη εγκυρότητας και την εξουσία για την επιβολή περιορισμών. Με το blockchain να λειτουργεί σαν ένας μηχανισμός συναίνεσης για να εξασφαλίσει ότι οι κόμβοι είναι σε συγχρονισμό μεταξύ τους, οι συναλλαγές μπορούν να επαληθευτούν και να επεξεργαστούν από κάθε κόμβο ανεξάρτητα. Η αποκεντρικοποίηση είναι επίσης καλή διότι αν τα δεδομένα της ήταν αποθηκευμένα σε ένα υπολογιστή κάποιου τρίτου, όπως οι τράπεζες ή η κυβέρνηση, κάποιος κακόβουλος που με κάποιο τρόπο έπαιρνε πρόσβαση στο σύστημα θα μπορούσε να τροποποιήσει τα δεδομένα. Ως εκ τούτου αυτοί οι οργανισμοί χρειάζονται να προσλάβουν ανθρώπους και να σχεδιάσουν μεθόδους για να αποτρέψουν την αλλοίωση των δεδομένων. Αυτό απαιτεί πολύ χρόνο αλλά και χρήμα,[23] αυτό το πρόβλημα δεν υπάρχει στο blockchain επειδή όπως είπαμε η βάση δεδομένων είναι κατανεμημένοι σε όλους του κόμβους του δίκτυο και για να γίνει κάποια αλλαγή πρέπει να υπάρχει συναίνεση μεταξύ των κόμβων.

Έλεγχος από τους χρήστες : Στο blockchain οι χρήστες έχουν τον έλεγχο των πληροφοριών τους και των συναλλαγών τους.[23] Σε μία κεντρικοποιημένη εφαρμογή είναι δυνατή η αποθήκευση των προσωπικών δεδομένων κάθε χρήστη καθώς και οι ηλεκτρονικές τους συνήθειες, όπως για παράδειγμα το Facebook και η Google πράγμα που δεν ισχύει στο blockchain.[29]

Ποιοτικά δεδομένα : Το blockchain λόγω των αυστηρών κριτηρίων για την επικύρωση των συναλλαγών, όπως αναλύθηκε στο υπό κεφάλαιο 3.3.2, φιλτράρει κακά ή ψευδή δεδομένα,[4] Αυτό έχει σαν αποτέλεσμα τα δεδομένα που καταγράφονται στο blockchain να είναι πλήρη, συνεπής, έγκαιρα, ακριβή και ευρέως διαθέσιμα.[23]

Ανθεκτικότητα, αξιοπιστία και μακροζωία : Λόγω του ότι το blockchain είναι ένα αποκεντρικοποιημένο σύστημα, δεν έχει ένα κεντρικό σημείο αποτυχία κάνοντας ανθεκτικό σε τεχνικά προβλήματα και κακόβουλες επιθέσεις. Επίσης λόγο ότι κάθε κόμβος του δικτύου έχει ένα αντίγραφο της βάσης δεδομένων, αν ένας ή περισσότεροι βγει εκτός σύνδεσης δεν θα επηρεαστεί η διαθεσιμότητα ή η ασφάλεια του δικτύου, κάνοντας το blockchain αξιόπιστο.[4] Επί πρόσθετα το blockchain θα υπάρχει όσο υπάρχουν και ενεργοί κόμβοι στο δίκτυο το οποίο αναμένεται να ισχύει για πολλά χρόνια ακόμα.

Ακεραιότητα της διαδικασίας : Οι συναλλαγές εκτελούνται ακριβώς όπως ορίζει το πρωτόκολλο, αφαιρώντας την ανάγκη για μεσάζοντα.[23]

Διαφάνεια και αμεταβλητότητα : Στο blockchain κανείς δεν μπορεί να τροποποιήσει η να διαγράψει τα δεδομένα που υπάρχουν στο blockchain, αυτό συμβαίνει επειδή, τροποποιώντας κάποιο μπλοκ αλλάζει τελείως το κρυπτογραφικό του αποτύπωμα[4], καθιστώντας το άκυρο, διότι δεν μπορεί να συνδεθεί με το γονικό του μπλοκ και ούτε με το παιδί του. Επίσης αν κάποιος προσπαθήσει να αλλάξει τα δεδομένα, θα το προσέξουν όλοι οι υπόλοιποι χρήστες επειδή, όλοι οι χρήστες μπορούν να δουν τα δεδομένα του blockchain.[4]

Απλοϊκό οικοσύστημα : Όλες οι συναλλαγές προθέτονται σε ένα εδάφιο, λύνοντας τα προβλήματα ακαταστασίας και επιπλοκών που μπορεί να δημιουργηθούν από την ύπαρξή πολλών εδαφίων.[23]

Γρηγορότερες συναλλαγές : Οι διατραπεζικές συναλλαγές μπορεί να διαρκέσουν μέρες μέχρι να ολοκληρωθούν πλήρως, ειδικά τις μη εργάσιμες μέρες, αντιθέτως οι συναλλαγές στο blockchain μπορούν να γίνουν μέσα σε μερικά λεπτά κάθε μέρα και κάθε ώρα.[23]

Χαμηλό κόστος συναλλαγών : Λόγο ότι δεν χρειάζονται οι μεσάζοντες, αποφεύγονται οι προμήθειες των συναλλαγών.[23]

2.7.2 Τα μειονεκτήματα

Παρόλο που η τεχνολογία blockchain έχει πολλά και σημαντικά πλεονεκτήματα, έχει και κάποια σημαντικά μειονεκτήματα που πρέπει να ληφθούν υπ' όψη.

Χαμηλή απόδοση: Το κύριο μειονέκτημα του blockchain είναι η υψηλή κατανάλωση ηλεκτρικής ενέργειας, όπως αναφέραμε προηγούμενος κάθε κόμβος και κάθε miner, έχει το αντίγραφο του blockchain, προσφέροντας ανοχή σε σφάλματα, μηδενικές διακοπές των υπηρεσιών και τα δεδομένα του απαράλλαχτα. Αυτό όμως έχει σαν αποτέλεσμα την τεράστια σπατάλη ηλεκτρικής ενέργειας, λόγο ότι οι κόμβοι είναι σε συνεχή λειτουργία,[14] καθώς επίσης επειδή επαναλαμβάνει κάθε κόμβος την διαδικασία της επαλήθευσης των ψηφιακών υπογραφών κάθε συναλλαγής και μέσω της διαδικασία συναίνεσης όπως το PoW, οι οποίες είναι και οι δύο υπολογιστικά περίπλοκες και χρειάζονται μεγάλή χρήση υπολογιστικής ισχύς. Άλλος ένας λόγος είναι επίσης και ο πλεονασμός, καθώς οι παραπάνω διαδικασίες γίνονται ανεξάρτητα από πολλούς υπολογιστές και στην περίπτωση του PoW υπάρχει μόνο ένας νικητής κόμβος κάθε φορά, με αποτέλεσμα η υπολογιστική προσπάθεια των υπολοίπων κόμβων να πάει χαμένη.[23]

Τροποποίηση δεδομένων : Παρόλο που η τροποποίηση των δεδομένων είναι σχεδόν αδύνατη είναι ένα πλεονέκτημα για την αποφυγή κακόβουλων ενεργειών, είναι ταυτόχρονα μειονέκτημα σε περίπτωση που χρειάζεται η αλλαγή κάποιου δεδομένου ή κώδικα στο blockchain. Για να γίνει μια τέτοια αλλαγή χρειάζεται μια εξαναγκασμένη διακλάδωση η οποία είναι πολύ απαιτητική σε πόρους διαδικασία.[4]

Ιδιωτικό κλειδί : Το blockchain χρησιμοποιεί ασύμμετρη κρυπτογραφία για να δώσει τον έλεγχο σε κάποιον για τα κρυπτό νομίσματα του ή άλλα δεδομένα στο blockchain. Κάθε διεύθυνση στο blockchain έχει κάποιο αντίστοιχο ιδιωτικό κλειδί και ενώ μπορεί κανείς να μοιραστεί την διεύθυνση του, το ιδιωτικό του κλειδί πρέπει να παραμείνει μυστικό. Αυτό το κλειδί χρησιμοποιείται για την πρόσβαση στα χρήματα του, το οποίο σημαίνει ότι κάθε χρήστης λειτουργεί σαν την προσωπική του τράπεζα. Αν κάποιος χάσει το κλειδί του, χάνει και τα χρήματα του και δεν μπορεί να κάνει τίποτα γι' αυτό.[4]

Διαφάνεια : Ένα ακόμα πλεονέκτημα που είναι επίσης μειονέκτημα, λόγο ότι όλες οι πληροφορίες σχετικά με τις συναλλαγές είναι δημόσιες, το blockchain δεν πρέπει να χρησιμοποιηθεί σε εφαρμογές που διαχειρίζονται ευαίσθητα δεδομένα.[4]

Προβλήματα ενσωμάτωσης : Οι εφαρμογές του blockchain προσφέρουν λύσεις που χρειάζονται σημαντικές αλλαγές ή και γενική αντικατάσταση των υπαρχόντων συστημάτων. Για να κάνουν αυτήν την αλλαγή, οι εταιρίες πρέπει να την προγραμματίσουν προσεκτικά.[23]

2.7.3 Οι απειλές

Το blockchain απειλείται από διάφορους κινδύνους περισσότερο θεωρητικά διότι είναι δύσκολο να πραγματοποιηθούν στην πράξη.

51% επίθεση : Το blockchain βασίζεται στους μηχανισμούς κατανεμημένης συναίνεσης για να διατηρηθεί η αμοιβαία εμπιστοσύνη στο δίκτυο. Όμως οι μηχανισμοί συναίνεσης έχουν μία αδυναμία την οποία οι επιτιθέμενοι μπορούν να αποκτήσουν τον έλεγχο ολόκληρου του δικτύου.

Το blockchain έχει σχεδιαστεί με την υπόθεση ότι όλοι στο δίκτυο είναι καλόβουλοι, αν όμως κάποιος ή μία ομάδα χρηστών καταφέρουν να πάρουν στον έλεγχο τους πάνω από το 50% της συνολικής υπολογιστικής ισχύς κατακερματισμού στον PoW τότε είναι δυνατή μια 51% επίθεση. Αυτή η επίθεση θεωρείται και η πιο απειλητική διότι δίνει στο επιτιθέμενο την δύναμη να καταστρέψει την σταθερότητα του δικτύου μέσο double spending, εξαίρεση, τροποποίηση των συναλλαγών και να αποτρέψει την εξόρυξη μερικών ή και όλων των έγκυρων μπλοκ προς όφελος του.[28]

Double spending : Το double spending είναι μία επίθεση όπου κάποιος κακόβουλος χρήστης προσπαθεί να ξοδέψει το ίδιο κρυπτό νόμισμα πολλαπλές φορές. Πρακτικά αυτό είναι σχεδόν αδύνατο λόγο ότι κάθε συναλλαγή επαληθεύεται και αν υπάρξει άλλη συναλλαγή με ίδια στοιχεία γίνεται αυτομάτως άκυρη.[28]

Σπάσιμο της κρυπτογραφίας : Είναι δυνατό η κρυπτογραφία που χρησιμοποιεί το blockchain να σπάσει με την χρήση κβαντικών αλγόριθμων και της αντίστροφής μηχανικής για να βρεθεί το ιδιωτικό κλειδί του στόχου. Για την επίλυση αυτού του πιθανού κινδύνου, προτείνεται η χρήση κβαντικής κρυπτογραφίας.[33]

3 ΚΕΦΑΛΑΙΟ 3ο : Ethereum

Το Ethereum είναι ένα δίκτυο peer-to-peer από εικονικές μηχανές (virtual machines) όπου κάθε developer μπορεί να χρησιμοποιήσει για να «τρέξει» κατανεμημένες εφαρμογές (DApps) και χρησιμοποιεί σαν νόμισμα το ether. Αυτά τα προγράμματα μπορούν να είναι το οτιδήποτε, αλλά το δίκτυο είναι φτιαγμένο έτσι ώστε να τα εκτελεί όταν εκπληρωθούν οι προϋποθέσεις, όπως σε κάποιο συμβόλαιο. Το Ethereum χρησιμοποιεί ένα αποκεντρικοποιημένο και δημόσιο Blockchain όπου αποθηκεύει, εκτελεί και προστατεύει αυτά τα συμβόλαια με κρυπτογραφικό τρόπο. Κάθε υπολογιστής στο δίκτυο κατεβάζει ένα μικρό virtual machine για να συγχρονιστεί στο Blockchain του Ethereum και παραμένει διαθέσιμος για την εκτέλεση συμβολαίων. Αυτό το κατανεμημένο δίκτυο υπολογιστών προσφέρει ασφάλεια, αξιοπιστία και την υπολογιστική ισχύ που χρειάζονται οι εφαρμογές.[19]

3.1 Γιατί Ethereum

Το Ethereum ως σύστημα που χρησιμοποιεί το blockchain μοιράζεται τα πλεονεκτήματα του, όμως ανάλογος την εφαρμογή έχει πλεονεκτήματα που το κάνουν να υπερτερεί έναντι στο blockchain του Bitcoin. Τα πλεονεκτήματα αυτά είναι :

- **Γρηγορότερες συναλλαγές** : Δεν υπάρχει όριο μεγέθους στο Blockchain του Ethereum, ο αριθμός των συναλλαγών σε ένα μπλοκ εξαρτάται από την δουλειά των miner. Προς το παρόν ένα μπλοκ στο Ethereum ανακαλύπτεται κάθε 10 με 20 δευτερόλεπτα.[1]
- **Είναι κάτι παραπάνω από ένα νόμισμα** : Παρόλο που το Bitcoin και το Ethereum είναι και τα δύο ψηφιακά νομίσματα, ο σκοπός του Ethereum δεν είναι να γίνει ένας εναλλακτικός τρόπος πληρωμής όπως το Bitcoin, αλλά η ανάπτυξη κατανεμημένων εφαρμογών.[1]
- **Πληρότητα Turing** : Το Ethereum είναι φτιαγμένο με κώδικά που είναι Turing πλήρης. Ένα λογισμικό ή πρόγραμμα θεωρείται Turing πλήρες εάν μπορεί να τρέξει οποιοδήποτε κώδικα, με την προϋπόθεση ότι έχει αρκετούς πόρους συστήματος. Αυτό αφαιρεί την ανάγκη για συγκεκριμένο λογισμικό ή υπολογιστές που τρέχουν μόνο συγκεκριμένους κώδικες.[1]
- **Πλούσιο σε καταστάσεις** : Το Ethereum έχει την δυνατότητα να θυμάται και να διατηρεί περισσότερες καταστάσεις σε επίπεδο Blockchain. Το Bitcoin δεν έχει καταστάσεις διότι μπορεί μόνο να πραγματοποιεί συναλλαγές, αντιθέτως το Ethereum, εκτός από το να διατηρεί χρηματικό υπόλοιπο, μπορεί να χειρίζεται κώδικα συμβολαίων και δεδομένα.[1]

Το αποτέλεσμα είναι η δημιουργία ενός εναλλακτικού πρωτόκολλου για την υλοποίηση αποκεντρικοποιημένων εφαρμογών, με έμφαση σε περιπτώσεις όπου η ταχεία ανάπτυξη, η ασφάλεια για μικρές και σπάνια χρησιμοποιημένες εφαρμογές και η ικανότητα διαφορετικών εφαρμογών να αλληλοεπιδρούν μεταξύ τους, είναι σημαντική. Το Ethereum με την χρήση ενός blockchain που υποστηρίζει μία γλώσσα προγραμματισμού με πληρότητα Turing, επιτρέπει στον καθένα να φτιάξει έξυπνα συμβόλαια και αποκεντρικοποιημένες εφαρμογές όπου μπορούν να δημιουργήσουν αυθαίρετους κανόνες για ιδιοκτησία των έξυπνων συμβολαίων, για διάφορες μορφές συναλλαγών και απλές έως και περίπλοκες συναρτήσεις που προκαλούν μεταβολές στα δεδομένα.[19]

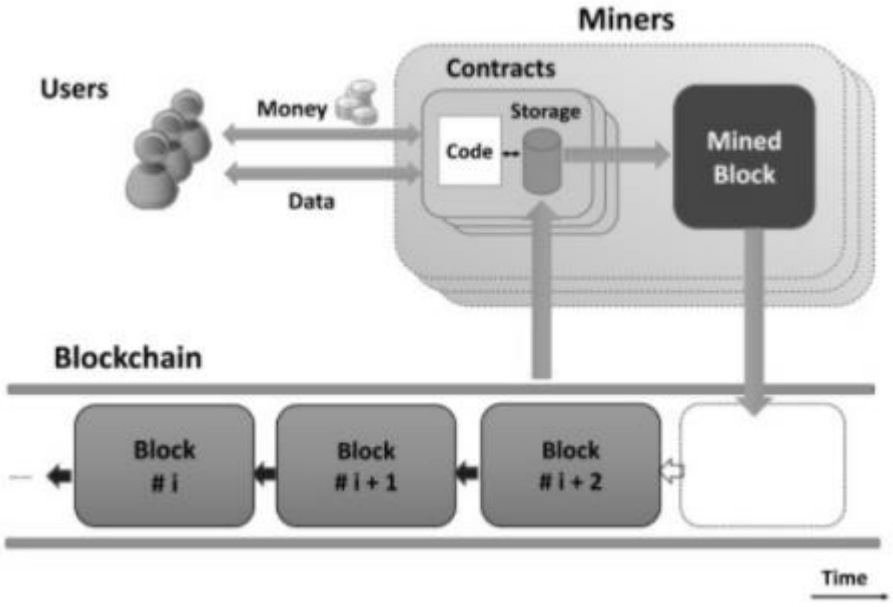
3.2 EVM (Ethereum Virtual Machine)

Το EVM είναι το κομμάτι του Ethereum που διαχειρίζεται την ανάπτυξη και την εκτέλεση έξυπνων συμβολαίων στο δίκτυο του Ethereum. Ουσιαστικά το EVM είναι ένας παγκόσμιος και αποκεντρικοποιημένος υπολογιστής, κατανεμημένος σε όλους τους υπολογιστές που συμμετέχουν στο δίκτυο του Ethereum, ο οποίος περιέχει εκατομμύρια εκτελέσιμα αντικείμενα τα οποία έχουν το καθένα τον δικό του μόνιμο αποθηκευτικό χώρο. Αυτός ο παγκόσμιος υπολογιστής εκτελεί οποιαδήποτε υπολογιστική διεργασία η οποία αλλάζει την κατάσταση ενός δεδομένου εκτός από απλές συναλλαγές μεταξύ λογαριασμών. Το EVM είναι μία σχεδόν Turing πλήρης μηχανή καταστάσεων λόγο ότι υπάρχει περιορισμός των υπολογιστικών βημάτων στις εκτελέσιμες διεργασίες από το ποσό του gas που είναι διαθέσιμο για την εκτέλεση οποιουδήποτε έξυπνου συμβολαίου.[3]

3.3 Smart contracts

Τα Έξυπνα συμβόλαια (smart contracts) είναι εκτελέσιμος κώδικας που τρέχει στο blockchain για την εκτέλεση και την επιβολή των όρων μίας συμφωνίας. Σκοπός των έξυπνων συμβολαίων είναι η αυτόματη εκτέλεση των όρων μία συμφωνίας όταν πληρούνται οι καθορισμένες προϋποθέσεις, ως εκ τούτου τα έξυπνα συμβόλαια προσφέρουν χαμηλά τέλη συναλλαγής σε σύγκριση με τα παραδοσιακά συστήματα τα οποία χρειάζονται κάποιον ενδιάμεσο φορέα για την εκτέλεση των όρων της συμφωνίας.[3]

Ένα έξυπνο συμβόλαιο αποτελείται από το υπόλοιπο του λογαριασμού, έναν ιδιωτικό αποθηκευτικό χώρο και τον εκτελέσιμο κώδικα. Η κατάσταση του συμβολαίου αποτελείται από την αποθηκευτικό χώρο και το υπόλοιπο, αυτή η κατάσταση είναι αποθηκευμένη στο blockchain και εκσυγχρονίζεται κάθε φορά που καλείται το συμβόλαιο από κάποιον χρήστη. Στην εικόνα 3.1 μπορούμε να δούμε πώς αλληλοεπιδρά ένα έξυπνο συμβόλαιο με τους χρήστες και με το blockchain.[3]



Πίνακας 3.1 Αλληλεπίδραση έξυπνου συμβολαίου με τους χρήστες και το blockchain[3]

3.3.1 Ο κύκλος ζωής ενός έξυπνου συμβολαίου

Τα έξυπνα συμβόλαια γράφονται σε γλώσσες υψηλού επιπέδου, όπως η solidity. Όμως για να τρέξουν πρέπει να μεταγλωττιστούν σε χαμηλού επιπέδου bytecode που τρέχει στο Ethereum Virtual Machine. Μόλις γίνει η μεταγλώττιση, γίνεται η ανάπτυξη στην πλατφόρμα του Ethereum μέσω μίας ειδικής συναλλαγής για την δημιουργία του συμβολαίου. Κάθε συμβόλαιο αναγνωρίζεται από την Ethereum διεύθυνση της, η οποία δημιουργείται από την συναλλαγή δημιουργίας συμβολαίου ως συνάρτηση του λογαριασμού που την δημιούργησε και του αριθμού nonce. Αυτή η διεύθυνση μπορεί να χρησιμοποιηθεί στις συναλλαγές ως παραλήπτης, για την αποστολή χρημάτων στο συμβόλαιο ή για την κλήση κάποια από τις συναρτήσεις του συμβολαίου. Μία από τις ιδιαιτερότητες των διευθύνσεων των συμβολαίων είναι ότι σε αντίθεση με τους λογαριασμούς των χρηστών, δεν υπάρχουν κλειδιά συσχετισμένα με τον λογαριασμό που δημιουργήθηκε για το έξυπνο συμβόλαιο. Ο δημιουργός του συμβολαίου δεν έχει ειδικά δικαιώματα σε επίπεδο πρωτοκόλλου, μπορεί όμως να τα προσθέσει σαν κώδικα στο έξυπνο συμβόλαιο του, και ουσιαστικά ένας λογαριασμός έξυπνου συμβολαίου δεν έχει ιδιοκτήτη αλλά ανήκει στον εαυτό του.[3]

Επίσης τα έξυπνα συμβόλαια εκτελούνται μόνο αν κληθούν από κάποια συναλλαγή, η οποία έγινε από κάποιον χρήστη. Ένα συμβόλαιο μπορεί να καλέσει κάποιο άλλο και αυτό να καλέσει ένα τρίτο και ούτω καθεξής, δημιουργώντας μία αλυσίδα, όμως σ' αυτήν την αλυσίδα το πρώτο συμβόλαιο θα έχει κληθεί από κάποια συναλλαγή από κάποιον χρήστη. Κανένα συμβόλαιο δεν μπορεί να ξεκινήσει από μόνο του ούτε να τρέξει στο παρασκήνιο, παραμένουν αδρανείς μέχρι κάποια συναλλαγή προκαλέσει την εκτέλεση του είτε άμεσα είτε έμμεσα, επιπρόσθετα τα έξυπνα συμβόλαια δεν εκτελούνται παράλληλα αλλά σειριακά.[3]

Οι συναλλαγές είναι ατομικές, ανεξαρτήτως πόσα συμβόλαια καλούν ή το τι κάνουν αυτά τα συμβόλαια όταν καλεστούν και εκτελούνται πλήρως, με οποιαδήποτε αλλαγή στις καταστάσεις του συμβολαίου ή των λογαριασμών των χρηστών να καταγράφεται μόνο αν η εκτέλεση ολοκληρώθηκε επιτυχημένα χωρίς κανένα σφάλμα. Αν η εκτέλεση αποτύχει λόγο κάποιου σφάλματος, όλες οι αλλαγές που έγινα θα γυρίσουν στην αρχική τους κατάσταση, σαν να μην έγινε πότε αυτή η συναλλαγή. Αυτή η συναλλαγή θα καταγραφεί σαν αποτυχημένη προσπάθεια και τα ether που ξοδεύτηκαν για το gas αφαιρούνται από τον λογαριασμό που επιχείρησε την συναλλαγή.[3]

Όπως είπαμε ο κώδικας ενός συμβολαίου δεν μπορεί να αλλάξει, μπορεί όμως να διαγραφθεί το έξυπνο συμβόλαιο, με την αφαίρεση του κώδικα και των στοιχείων στο αποθηκευτικό του χώρο από την διεύθυνση του αφήνοντας έναν κενό λογαριασμό. Όλες οι συναλλαγές που αποστέλνονται σε αυτή την διεύθυνση δεν προκαλούν την εκτέλεση κάποιο κώδικα επειδή δεν υπάρχει πλέον κώδικας προς εκτέλεση. Για να διαγραφτεί ένα συμβόλαιο, εκτελείτε ένα opcode του EVM που λέγεται SELFDESTRUCT. Αυτή η λειτουργία κοστίζει αρνητικό gas, δηλαδή επιστρέφει ether, ενθαρρύνοντας έτσι την απελευθέρωση πόρων του δικτύου που χρησιμοποιούνται από κάποιο συμβόλαιο. Αυτή η διαγραφή δεν αφαιρεί το ιστορικό των συναλλαγών του συμβολαίου λόγο ότι το blockchain είναι αμετάβλητο. Είναι σημαντικό επίσης να αναφερθεί ότι η δυνατότητα της διαγραφής πρέπει να προγραμματιστεί στο συμβόλαιο από τον δημιουργό του, σε αντίθετη περίπτωση το συμβόλαιο θα παραμείνει για πάντα στο blockchain.[3]

3.3.2 Ασφάλεια των έξυπνων συμβολαίων και κενά ασφαλείας

Κατά την δημιουργία των έξυπνων συμβολαίων είναι πολύ σημαντικό να λαμβάνουμε υπόψη την ασφάλεια λόγο ότι τα λάθη στον κώδικα είναι δαπανηρά και μπορεί εύκολα να τα εκμεταλλευτεί κάποιος και οι ζημίες είναι αδύνατο να ανακτηθούν.

Όπως και με κάθε πρόγραμμα, ένα έξυπνο συμβόλαιο θα εκτελέσει ακριβώς αυτό που έχει προγραμματιστεί να κάνει, το οποίο δεν είναι πάντα αυτό που σκόπευε να φτιάξει ο προγραμματιστής. Επίσης τα έξυπνα συμβόλαια είναι δημόσια και κάθε χρήστης μπορεί να αλληλοεπιδράσει με αυτά δημιουργώντας απλά μία συναλλαγή, με αποτέλεσμα την πιθανή εκμετάλλευση κάποιου τρωτού σημείου του κώδικα από κάποιον κακόβουλο χρήστη.[28] Ως εκ τούτου είναι πολύ σημαντικός ο έλεγχος και η αποσφαλμάτωση των έξυπνων συμβολαίων πριν ανέβουν τελικά στο δίκτυο του Ethereum.[3]

Παρακάτω μπορούμε μερικά από τα κενά ασφαλείας που μπορούν να προκύψουν :

- **Επανεισδοχή(Reentrancy)** : Μία επίθεση επανεισδοχής μπορεί να προκύψει όταν έναdesυμβόλαιο καλεί ένα εξωτερικό συμβόλαιο το οποίο παίρνει τον έλεγχο της ροής και δημιουργεί ένα αίτημα πίσω στο συμβόλαιο που έκανε το αρχικό αίτημα πριν να τελειώσει η πρώτη επίκληση και να επιφέρει απρόβλεπτες συνέπειες.[18]
- **Απροστάτευτο SELFDESTRUCT** : Αυτό το κενό ασφαλείας προκύπτει από ένα λογικό σφάλμα στην Solidity. Ένα έξυπνος συμβόλαιο επιτρέπει την πρόσβαση σε ένα μη εξουσιοδοτημένο άτομο, όπως για παράδειγμα λόγο ελλιπή ή ανεπαρκή έλεγχο πρόσβασης, κάποιος κακόβουλος χρήστης μπορεί να καλέσει την συνάρτηση SELFDESTRUCT με αποτέλεσμα να καταστραφεί το συμβόλαιο και το υπόλοιπο που υπάρχει στο συμβόλαιο να μεταφερθεί σε αυτόν.[18]
- **Υπερχείλιση και Υποχείλιση ακέραιων** : μία υπερχείλιση ή υποχείλιση συμβαίνει όταν μία μαθηματική πράξη μόλις γίνει υπερβαίνει την μέγιστη ή αντίστοιχα την ελάχιστη τιμή που μπορεί να λάβει, της οποίας τα όρια εξαρτάται από τον τύπο της μεταβλητής, πχ για τύπο uint8 τα όρια είναι από 0 εως 255.[18]
- **Κλειδωμένα χρήματα** : Όταν ο δημιουργός ξεχνάει να εισάγει την διεύθυνση που θέλει να στείλει, πχ νομίσματα, το πρόγραμμα από προεπιλογή τα στέλνει στην διεύθυνση 0x0 όπου και μένουν εκεί για πάντα.[18]
- **Αίτημα εξουσιοδότησης (delegatecall)** : Αυτό το κενό ασφαλείας συμβαίνει όταν ένα έξυπνο συμβόλαιο κάνει ένα delegatecall σε ένα μη έμπιστο έξυπνο συμβόλαιο, κατά την αποστολή του αιτήματος τα περιεχόμενα, όπως το msg.sender, είναι τα ίδια με το αίτημα που έγινε στο πρώτο συμβόλαιο, έτσι λόγο αυτού ένα μη έμπιστο συμβόλαιο θα μπορεί κάνει αλλαγές προς το συμφέρον του στις αποθηκευμένες τιμές του πρώτου.[18]

Είναι δύσκολο να φτιαχτεί κώδικας χωρίς ελαττώματα, όμως ακολουθώντας κάποιες καλές πρακτικές μπορούμε να τα ελαχιστοποιήσουμε με τους παρακάτω τρόπους :

- **Απλότητα** : Η περιπλοκότητα είναι εχθρός της ασφάλειας. Όσο πιο απλός είναι ο κώδικας τόσο πιο λίγα κάνει ταυτόχρονα όμως μειώνει τις πιθανότητες για bugs ή απρόβλεπτες καταστάσεις.[3]
- **Επαναχρησιμοποίηση κώδικα** : Αν ήδη υπάρχει κάποια βιβλιοθήκη ή κάποιο συμβόλαιο το οποίο κάνει ένα κάποιο κομμάτι από το επιθυμητό αποτέλεσμα είναι προτιμότερη η χρήση του παρά να γραφτεί από την αρχή. Επίσης αν ένα κομμάτι κώδικα επαναλαμβάνεται πολλές φορές είναι καλύτερο να γραφτεί σαν συνάρτηση ή βιβλιοθήκη.[3]

- **Ποιότητα κώδικα** : Τα λάθη στα έξυπνά συμβόλαια πληρώνονται ακριβά, κάθε bug μπορεί να οδηγήσει σε απώλειες νομισμάτων. Η συγγραφή κώδικα solidity δεν είναι σαν την δημιουργία προγράμματος σε κάποια άλλη γλώσσα προγραμματισμού, χρειάζεται αυστηρή μηχανική και μεθόδους ανάπτυξης λογισμικού διότι μόλις ανέβει στο blockchain δεν υπάρχει εύκολος τρόπος να διορθωθούν τα προβλήματα.[3]
- **Αναγνωσιμότητα και δυνατότητα ελέγχου** : Ο κώδικας πρέπει να είναι καθαρός και εύκολος στην κατανόηση. Όσο πιο εύκολος στην ανάγνωση είναι ο κώδικας τόσο πιο εύκολος είναι και ο έλεγχος. Τα έξυπνα συμβόλαια είναι δημόσια και όλοι μπορούν να τον διαβάσουν καθώς και να εφαρμόσουν αντίστροφη μηχανική, γι' αυτό είναι προτιμότερο η ανάπτυξη να γίνεται δημόσια χρησιμοποιώντας συνεργατικές και ανοιχτού κώδικα μεθόδους, για να γίνεται χρήση της συσσωρευμένης γνώσης της προγραμματιστικής κοινότητας.[3]
- **Έλεγχος** : Στο blockchain ο καθένας μπορεί να τρέξει ένα έξυπνο συμβόλαιο με οποιαδήποτε παράμετρο εισόδου θέλει, γι' αυτό είναι πολύ σημαντικό να γίνεται εκτενείς έλεγχος σε όλες πιθανές παραμέτρους για να γίνει σίγουρο ότι τα αποτελέσματα είναι τα αναμενόμενα.[3]

3.3.3 Αναβαθμίσεις στα έξυπνα συμβόλαια

Παρόλο που είναι δύσκολες οι αλλαγές στα έξυπνα συμβόλαια υπάρχουν μερικές τεχνικές για να γίνεται αναβάθμιση του κώδικα.

Συμβόλαια αφέντη-σκλάβου. Αυτή είναι η πιο απλή και εύκολη στην κατανόηση τεχνική για να υπάρχει δυνατότητα αναβάθμισης των συμβολαίων. Σε αυτήν την μέθοδο, δημιουργείται ένα συμβόλαιο αφέντη το οποίο περιέχει τις διευθύνσεις όλων των συμβολαίων σκλάβων, και τα συμβόλαια σκλάβοι όταν θέλουν να επικοινωνήσουν με κάποιο συμβόλαιο λαμβάνουν την διεύθυνση του από το συμβόλαιο αφέντη. Οπότε αν χρειαστεί να γίνει κάποια αναβάθμιση σε κάποιο συμβόλαιο το μόνο που χρειάζεται είναι να ανεβάσουμε το καινούργιο συμβόλαιο στο δίκτυο και να αλλάξουμε την διεύθυνση στο συμβόλαιο αφέντη ώστε να δείχνει στο νέο. Ένας από τους περιορισμούς αυτής την τεχνικής είναι ότι δεν είναι δυνατή η εύκολη μεταφορά των δεδομένων του συμβολαίου στο καινούργιο.[16]

Συμβόλαια αιώνιου αποθηκευτικού χώρου. Με αυτήν την τεχνική χωρίζουμε τα συμβόλαια που περιέχουν την λογική και τα δεδομένα. Το συμβόλαιο με τα δεδομένα θα είναι μόνιμο και μη αναβαθμίσιμο, εκτός από την ενημέρωση για νέο συμβόλαιο λογικής, ενώ το συμβόλαιο με την λογική μπορεί να αναβαθμιστεί όσες φορές χρειάζεται. Το μειονέκτημα αυτής της τεχνικής είναι ότι επειδή το συμβόλαιο του αποθηκευτικού χώρου δεν έχει δυνατότητα αναβάθμισης, οποιαδήποτε ανάγκη για αλλαγή στην δομή των δεδομένων ή κάποιο bug στα δεδομένα, μπορούν να καταστήσουν όλα τα δεδομένα του συμβολαίου άχρηστα. Επίσης ένα άλλο πρόβλημα είναι ότι το συμβόλαιο με την λογική χρειάζεται να κάνει εξωτερικά αιτήματα άμα θέλει να αποκτήσει πρόσβαση ή να επεξεργαστεί τα δεδομένα και τα εξωτερικά αιτήματα κοστίζουν επιπλέον gas. Αυτή η τεχνική συνήθως συνδυάζεται με την τεχνική συμβολαίων αφέντη-σκλάβου.[16]

Διακομιστής μεσολάβησης συμβολαίων αναβαθμιζόμενων αποθηκευτικών χώρων. Είναι δυνατό να αποφευχθεί το επιπλέον gas κάνοντας τα συμβόλαια αιώνιου αποθηκευτικού χώρου να λειτουργούν σαν διακομιστές μεσολάβησης στα συμβόλαια λογικής. Το συμβόλαιο μεσολάβησης και το συμβόλαιο λογικής θα έχουν κοινό ένα κοινό συμβόλαιο αποθηκευτικού χώρου έτσι ώστε η θέση αναφοράς του αποθηκευτικού χώρου να είναι κοινή στο EVM. Το συμβόλαιο μεσολάβησης θα έχει

μία συνάρτηση η οποία θα στείλει ένα αίτημα εξουσιοδότησης στο συμβόλαιο λογικής έτσι ώστε αυτό να μπορέσει να κάνει αλλαγές στον αποθηκευτικό χώρο του συμβολαίου μεσολάβησης.[16]

3.3.4 Εφαρμογές των έξυπνων συμβολαίων

Τα έξυπνα συμβόλαια λόγω των ιδιοτήτων τους έχουν ένα μεγάλο εύρος επιστημονικών πεδίων πού μπορούν να εφαρμοστούν τα οποία θα δούμε παρακάτω :

- **Τραπεζικό τομέα** : Ο τραπεζικός τομέας είναι ο πιο υποσχόμενος τομέας όπως μπορούν να εφαρμοστούν τα έξυπνα συμβόλαιο σε παραδοσιακές συναλλαγές όπως η πληρωμή δανείων και άλλες οικονομικές λειτουργίες.[31]
- **Συστήματα υγείας** : Τα έξυπνα συμβόλαια είναι επίσης χρήσιμα στον τομέα της υγεία. Απλοποιούν διάφορες λειτουργίες όπως την ανάκτηση των πληροφοριών ενός ασθενή, επιβεβαίωση δεδομένων και την έγκριση τους.[31]
- **Αλυσίδα εφοδιασμού και διαχείριση επιχείρησης** : Άλλος ένας τομέας εφαρμογής είναι η αλυσίδα εφοδιασμού, όπου τα έξυπνα συμβόλαια βοηθούν στην ομαλή παρακολούθηση των αποθεμάτων και των αγαθών και ελαχιστοποιούν τον κίνδυνο απάτης. Όσων αφορά την διαχείριση επιχειρήσεων, μπορούν να χρησιμοποιηθούν σαν συστήματα αυτόματης μίσθωσης των υπαλλήλων. [31]
- **Νομικά ζητήματα και ακίνητα** : Τα έξυπνα συμβόλαια λόγο της ουδετερότητας τους και της αυτοματοποίησης τους λειτουργούν καταλυτικά σε νομικά ζητήματα σε σύγκριση των παραδοσιακών μεθόδων πιστοποίησης των εγγράφων και των συμβολαιογραφικών πράξεων.
- **Κυβέρνηση** : Ο πιο υποσχόμενος αντίκτυπος της αποκεντροποίησης των έξυπνων συμβολαίων είναι σε κυβερνητικά συστήματα όπως η ψηφοφορία, λόγο ότι είναι αμετάβλητες οι ψήφοι και κανείς δεν μπορεί να παραποιήσει το αποτέλεσμα.[31]
- **Διαδίκτυο των πραγμάτων(Internet of Things)** : Το διαδίκτυο των πραγμάτων είναι ένας υποσχόμενος τομέας για έρευνα. Προς το παρόν γίνεται έρευνα για εφαρμογές με βάση την τεχνολογία blockchain όπως τα έξυπνα σπίτια, έξυπνες πόλης, έξυπνα μέσα μεταφορά και έξυπνη παρακολούθηση του περιβάλλοντος. Αν τα έξυπνα συμβόλαια ενσωματωθούν στα IoT συστήματα θα τα βοηθήσει να γίνουν πιο αυτόνομα.[21]
- **Διαχείριση ψηφιακών δικαιωμάτων** : Η βιομηχανία των ψηφιακών δικαιωμάτων εμπλέκει πολλούς συμβαλλόμενους για τον προγραμματισμό μιας εκδήλωσης. Το ποσοστό της πληρωμής και τα πνευματικά δικαιώματα είναι κάποια από τα ζητήματα που προκύπτουν, με την χρήση συστημάτων έξυπνων συμβολαίων εγγυούνται ότι τα δικαιώματα θα πάνε σε σωστούς παραλήπτες.[21]

  Ασφάλιση : Τα παραδοσιακά συστήματα ασφάλισης πολλές φορές κάνουν πολύ καιρό για την διαδικασία για να δώσουν την αποζημίωση στον ασφαλισμένο, με τα έξυπνα συμβόλαια αυτή η διαδικασία μπορεί να γίνει απλή και γρήγορη.[21]

3.3.5 Πλεονεκτήματα και Μειονεκτήματα

Παρακάτω μπορούμε να δούμε τα πλεονεκτήματα των έξυπνων συμβολαίων.

- **Γρήγορα και ακριβή** : Τα έξυπνα συμβόλαια εκτελούν εύκολα και γρήγορα μια προκαθορισμένη συμφωνία σε υπολογιστικό κώδικα σε σύγκριση με τον παραδοσιακό γραφειοκρατικό τρόπο.[31]
- **Αξιόπιστα** : Τα έξυπνα συμβόλαια έχουν προκαθορισμένους κανόνες τα οποία η συναλλαγή εκτελεί αυτόματα. Καθώς επίσης και οι καταχωρήσεις των εγγραφών είναι αμετάβλητες, ασφαλείς και κατανεμημένες στους εμπλεκόμενους της διαδικασίας.[31]

- **Ασφαλή** : Λόγο της τεχνολογίας blockchain τα έξυπνα συμβόλαια είναι ασφαλή από κακόβουλες παραποιήσεις των δεδομένων των συναλλαγών.[31]
- **Οικονομικά** : Τα έξυπνα συμβόλαια δεν χρειάζονται ενδιάμεσους για την εκτέλεση της συναλλαγής λόγο ότι οι όροι της συμφωνίας υπάρχουν σε μορφή κώδικα και εκτελούνται αυτόματα.[31]

Καθώς και τα μειονεκτήματα.

- **Αμεταβλητότητα** : Όπως αναφέραμε και πριν, μόλις ανέβη ένα έξυπνο συμβόλαιο στο δίκτυο του blockchain δεν γίνεται να αλλάξει κάτι στον κώδικα του, το οποίο είναι κακό σε περίπτωση που υπάρχουν λογικά σφάλματα ή κενά ασφαλείας.[25]
- **Ζητήματα ιδιωτικότητας** : Η τεχνολογία blockchain έχει ως αποτέλεσμα την διανομή των έξυπνων συμβολαίων σε όλους τους κόμβους του δικτύου και λόγο ότι οι συναλλαγές που καταγράφονται στο blockchain γίνονται με χρήση κρυπτογραφίας, όλοι οι συμμετέχοντες στο δίκτυο είναι ανώνυμοι. Όμως δεν υπάρχει ανωνυμότητα στην εκτέλεση των συμβολαίων διότι όλες οι συναλλαγές και τα περιεχόμενα τους είναι ορατά από όλους του χρήστες.[25]
- **Νομικά ζητήματα** : Παραδοσιακά για την σύναψη ενός συμβολαίου πρέπει να πληρούνται κάποιες προϋποθέσεις για να είναι έγκυρο. Μερικά από τα βασικά χαρακτηριστικά ενός έγκυρου συμβολαίου είναι η προσφορά από κάποιον συμβαλλόμενο, ή αποδοχή από κάποιον άλλο, μία υπόσχεση, η νομική ικανότητα για αμοιβαιότητα και σε κάποια συμβόλαια γραπτά νομικά έγγραφα. Αυτά τα στοιχεία σε ένα συμβόλαιο είναι κρίσιμα και κάποια από αυτά δεν είναι εφαρμόσιμα στα έξυπνα συμβόλαια.[25]

3.4 Τοκεν

Τα τοκεν αντιπροσωπεύουν αντικείμενα είτε στον ψηφιακό είτε στον πραγματικό κόσμο και χρησιμοποιούνται σε πολλά διαφορετικά επιστημονικά πεδία όπως τα οικονομικά και τεχνολογία υπολογιστών. Στην καθημερινότητα τα τοκεν χρησιμοποιούνται για να αντιπροσωπεύσουν αντικείμενα αξίας, στην τεχνολογία υπολογιστών υπάρχουν διάφοροι τύποι τοκεν, ορισμένα σαν αντικείμενα τα οποία αντιπροσωπεύουν τα δικαιώματα για την εκτέλεση κάποιας λειτουργίας, όπως για παράδειγμα τα τοκεν πρόσβασης, τα οποία χρησιμοποιούνται για την ταυτοποίηση των χρηστών και τα δικαιώματα που έχει σε ένα υπολογιστικό σύστημα και άλλα.[6]

Όπως και στην καθημερινή ζωή έτσι και στον κόσμο του blockchain τα τοκεν χρησιμοποιούνται για να αντιπροσωπεύσουν κάτι που έχει αξία, όπως οικόπεδα, έργα τέχνης και νομίσματα. Η σημαντική διαφορά όμως είναι ότι τα τοκεν που υπάρχουν ψηφιακά στο blockchain λειτουργούν κρυπτογραφικά, το οποίο σημαίνει ότι παράγονται, προστατεύονται και μεταφέρονται με την χρήση κρυπτογραφικών πρωτοκόλλων. Αφού καταλάβουμε τι είναι ένα τοκεν θα μπορούσαμε να ορίσουμε την τοκενοποίηση (tokenization) ως την διαδικασία η οποία μετατρέπει ένα περιουσιακό στοιχείο σε ένα ψηφιακό τοκεν στο blockchain, το οποίο μπορεί να αντιστοιχηθεί με τον λογαριασμό κάποιου χρήστη δίνοντας του τα ιδιοκτησιακά δικαιώματα.[6]

Τα τοκεν μπορούν επίσης να αντιπροσωπεύσουν ψηφιακά συλλεκτικά αντικείμενα στο blockchain, τα οποίαονομάζονται crypto collectibles.

3.4.1 Είδη τοκεν και χρήσεις

Χάρη στην βοήθεια των έξυπνων συμβολαίων, το Ethereum έχει γίνει μία πλατφόρμα με τοκεν διάφορων ειδών όπως βοηθητικά τοκεν, τοκεν παιχνιδιών και άλλα.

3.4.1.1 Fungible τοκεν

Από οικονομικής άποψης η έννοια fungibility είναι η ανταλλαξιμότητα ενός περιουσιακού στοιχείου με ένα άλλο του ίδιου τύπου, για παράδειγμα ένα χαρτονόμισμα των 10 ευρώ μπορεί να ανταλλαχτεί με ένα άλλο ίδια αξίας ή συνδυασμούς νομισμάτων ίσης αξίας. Τα fungible τοκεν λειτουργούν με τον ίδιο τρόπο. Τα fungible τοκεν είναι δυσδιάκριτα το ένα απ' το άλλο, είναι πλήρως ανταλλάξιμα με τοκεν ίδιας αξίας και είναι διαιρετά σε μικρότερα κομμάτια με την ανάλογη αξία.[6]

3.4.1.2 Non-fungible τοκεν

Τα non-fungible τοκεν(NFT) είναι ακριβώς το αντίθετο, το κάθε ένα είναι μοναδικό και διαφορετικό από τα άλλα τοκεν του ίδιου τύπου ή της ίδιας κατηγορίας, είναι μη ανταλλάξιμα μεταξύ τους λόγο της μοναδικότητας τους και τα χαρακτηριστικά που έχει σαν αποτέλεσμα να έχουν διαφορετική αξία και είναι αδιαίρετα διότι αντιστοιχούνται σε μία μοναδική μονάδα, για παράδειγμα ένα έργο τέχνης.[6]

3.4.1.3 Stable τοκεν

Τα stable τοκεν είναι ένα είδος τοκεν του οποίου η αξία αντιστοιχίζεται στην αξία ενός άλλου περιουσιακού στοιχείου όπως παραστατικό χρήμα ή πολύτιμου μετάλλου. Τα κρυπτό νομίσματα έχουν ασταθή αξία, το οποίο ελάττωμα τα κάνει ακατάλληλα για καθημερινή χρήση, τα stable τοκεν λύνουν αυτό το πρόβλημα λόγο ότι διατηρούν την αξία τους ίδια με αυτή του αντικειμένου που έχουν αντιστοιχηθεί.[6]

3.4.1.4 Security τοκεν

Η αξία των security τοκεν καθορίζεται από ανταλλάξιμα περιουσιακά στοιχεία όπως για παράδειγμα, τα security τοκεν μπορούν να αντιπροσωπεύσουν τις μετοχές σε μία εταιρία. Η διαφορά είναι ότι τα παραδοσιακά αξίογραφα κρατούνται στην τράπεζα και συναλλάζονται σε αγορές, ενώ τα security τοκεν είναι διαθέσιμα στο blockchain. Τα security τοκεν υπόκεινται σε όλους τους παραδοσιακούς νόμους και κανονισμούς όπως τα κανονικά αξιόγραφα.[6]

3.4.1.5 Χρήσεις των τοκεν

Τα τοκεν μπορούν να χρησιμοποιηθούν ως :

- **Νόμισμα** : Τα τοκεν μπορούν να χρησιμοποιηθούν ως νόμισμα του οποίου η αξία μπορεί να καθοριστεί μέσω ιδιωτικής συναλλαγής.[3]
- **Πόρους** : Τα τοκεν μπορούν να αντιπροσωπεύσουν έναν πόρο που απονεμήθηκε ή παράχθηκε σε ένα περιβάλλον κατανομής πόρων, όπως αποθηκευτικοί χώροι ή πόρους επεξεργαστή, αντιπροσωπεύοντας πόρους που μπορούν να διαμοιραστούν διαμέσου του δικτύου.[3]
- **Περιουσιακά στοιχεία** : Ένα τοκεν μπορεί να αντιπροσωπεύσει οποιοδήποτε περιουσιακό στοιχείο όπως χρήματα, οικόπεδα, αυτοκίνητα, αντικείμενα σε βίντεο παιχνίδια και άλλα.[3]
- **Πρόσβαση** : Ένα τοκεν μπορεί να αντιπροσωπεύσει τα δικαιώματα εισόδου και να παρέχουν πρόσβαση σε μία ψηφιακή ή φυσική ιδιοκτησία όπως φόρουμ, σε ένα δωμάτιο ξενοδοχείου ή την ενοικίαση αυτοκινήτου.[3]
- **Μετοχικό κεφάλαιο** : Ένα τοκεν μπορεί να αντιπροσωπεύσει το μερίδιο που έχει κάποιος σε μία ψηφιακή φυσική εταιρία.[3]
- **Ψήφο** : Ένα τοκεν μπορεί να αντιπροσωπεύσει το δικαίωμα ψήφου σε ένα ψηφιακό ή νομικό σύστημα.[3]

- **Συλλεκτικά** : Τα τοκεν μπορούν αντιπροσωπεύσουν ψηφιακά συλλεκτικά αντικείμενα όπως τα cryptokitties ή φυσικά όπως έργα τέχνης.[3]
- **Ταυτότητα** : Τα τοκεν μπορούν να αντιπροσωπεύσουν μία ψηφιακή ταυτότητα π.χ. avatar σε φόρουμ ή νομική ταυτότητα όπως την αστυνομική.[3]
- **Βεβαίωση** : Τα τοκεν μπορούν να αντιπροσωπεύσουν μία πιστοποίηση ή βεβαίωση κάποιου γεγονότος από κάποιο εξουσιοδοτημένο φορέα ή από ένα αποκεντρικοποιημένο σύστημα υπόληψης, όπως την καταγραφή ενός γάμου, ένα πιστοποιητικό γεννήσεως ή πτυχίο πανεπιστημίου.[3]
- **Utility** : Τα τοκεν μπορούν να χρησιμοποιηθούν για την πρόσβαση ή την πληρωμή μίας υπηρεσίας.[3]

Συνήθως ένα τοκεν μπορεί να χρησιμοποιηθεί ταυτόχρονα για περισσότερες εφαρμογές, μερικές φορές είναι δύσκολο να τα ξεχωρίσουμε, διότι τα αντίστοιχα φυσικά αντικείμενα έχουν συνδεθεί μεταξύ τους, όπως για παράδειγμα το δίπλωμα οδήγησης μπορεί να χρησιμοποιηθεί ως έγγραφο ταυτοποίησης.[3]

3.4.2 Πρότυπα των τοκεν

Τα πρότυπα των τοκεν είναι οι ελάχιστες προδιαγραφές για την υλοποίηση ενός τοκεν και επιτρέπουν στις εφαρμογές όπως τα ηλεκτρονικά πορτοφόλια να αναγνωρίζουν τα τοκεν και να αλληλοεπιδρούν με αυτά. Η κοινότητα συνεχώς συζητά και καθιερώνει πρότυπα για τοκεν στην γλώσσα προγραμματισμού solidity.[11] Θα δούμε μερικά βασικά πρότυπα παρακάτω.

Το πρότυπο ERC-20 είναι το πιο ευρέως χρησιμοποιημένο και το πιο γενικό πρότυπο το οποίο παρέχει τις βασικές λειτουργίες για την μεταφορά τοκεν καθώς επίσης επιτρέπει την δυνατότητα έγκρισης άλλου λογαριασμού έτσι ώστε να μπορεί να τα χρησιμοποιήσει κάποιος τρίτος.[11]

Το πρότυπο ERC-721 ή αλλιώς το non-fungible token standard, αφορά τοκεν τα οποία είναι διακριτά και επιτρέπει την παρακολούθηση περιουσιακών στοιχείων. Σε αυτό το πρότυπο πρέπει κάθε στοιχείο να μπορεί να αναζητηθεί σαν μεμονωμένο αντικείμενο.[11]

Το πρότυπο ERC-1155 ή αλλιώς πρότυπο πολλαπλών τοκεν επιτρέπει την διαχείριση οποιοδήποτε συνδυασμού fungible και non-fungible τοκεν σε ένα συμβόλαιο, περιλαμβάνοντας την μεταφορά πολλαπλών τοκεν με την μία.[11]

3.4.3 Πλεονεκτήματα και μειονεκτήματα των τοκεν

Τα τοκεν μοιράζονται πολλά από τα πλεονεκτήματα, όπως ταχύτερες συναλλαγές και διαφάνεια, και τα μειονεκτήματα του blockchain. Παρακάτω θα δούμε αυτά που αφορούν τα τοκεν.

Τα πλεονεκτήματα των τοκεν είναι εξής :

- **Ευελιξία** : Λόγο της παγκόσμιας υιοθέτησης συστημάτων που χρησιμοποιούν τοκεν, όπως συστήματα πληρωμών, η τοκενοποίηση γίνεται πιο απλή για χρήση διασυνοριακά.[6]
- **Χαμηλό κόστος** : Σε σύγκριση με τα παραδοσιακά χρηματοοικονομικά προϊόντα, η τοκενοποίηση έχει μικρότερο κόστος για την υλοποίηση, το οποίο έχει σαν αποτέλεσμα να επιφέρει χαμηλότερο κόστος στον τελικό χρήστη λόγο της ψηφιοποίησης.[6]
- **Αποκεντρικοποίηση** : Τα τοκεν υπάρχουν στο δημόσιο blockchain, εκμεταλλευόμενα την αποκεντρικοποίηση που προσφέρει η τεχνολογία blockchain. Όμως σε κάποιες περιπτώσεις είναι αποδεκτή λίγη κεντρικοποίηση λόγο του ελέγχου που χρειάζεται από τους επενδυτές και άλλων τρίτον που εμπλέκονται.[6]

- **Ασφάλεια** : Τα τοκεν προστατεύονται με κρυπτογραφικές μεθόδους και παράγονται χρησιμοποιώντας το blockchain. Αυτό έχει σαν αποτέλεσμα να είναι ασφαλή με την προϋπόθεση ότι κατά την υλοποίηση χρησιμοποιήθηκαν καλές πρακτικές, έτσι ώστε να αποφευχθούν κενά ασφαλείας τα οποία μπορούν να εκμεταλλευτούν οι χακερ.[6]
- **Τμηματική ιδιοκτησία** : Ένα περιουσιακό στοιχείο θα ήταν σχεδόν αδύνατο να έχει πολλαπλούς ιδιοκτήτες χωρίς νομικές, χρηματοοικονομικές και λειτουργικές προκλήσεις .Είναι δυνατό όμως, με την τοκενοποίηση, ένα περιουσιακό στοιχείο να ανήκει σε πολλούς κατόχους, όπως για παράδειγμα ένα οικόπεδο, γρηγορότερα, αποτελεσματικότερα, με ασφάλεια και λιγότερο περίπλοκο από τις παραδοσιακές μεθόδους.[6]
- **Εύκολη είσοδο** : Τα παραδοσιακά χρηματοοικονομικά συστήματα χρειάζονται και παραδοσιακούς τρόπους επαλήθευσης, η οποίες παίρνουν πολύ χρόνο, όμως στο blockchain δεν χρειάζεται όλη αυτή η διαδικασία επαλήθευσης, λόγο των κρυπτογραφικών εγγυήσεων που προσφέρει το blockchain και επίσης λόγο της φύσης των DApps το μόνο που χρειάζεται είναι η σύνδεση του λογαριασμού στην εφαρμογή μέσο ενός ψηφιακού πορτοφολιού και αν είναι αναγκαία η κατάθεση χρημάτων.[6]
- **Καινοτόμες εφαρμογές** : Η τοκενοποίηση είχε σαν αποτέλεσμα πολλές καινοτόμες εφαρμογές όπως ο συστήματα δανεισμού βιβλίων, ασφάλεια (ασφαλιστικών εταιριών), άλλες πολλές χρηματοοικονομικές εφαρμογές, όπως οι αποκεντρικοποιημένες συναλλαγές.[6]
- **Ρευστότητα** : Η ρευστότητα οφείλεται στην εύκολη διαθεσιμότητα των τοκεν από τους πολίτες. Με την χρήση των τοκεν, ακόμα και μη ρευστά περιουσιακά στοιχεία μπορούν να τοκενοποιηθούν και να πουληθούν σε κάποια αγορά.[6]

Παρόλο των πλεονεκτημάτων, υπάρχουν και αρκετά μειονεκτήματα που θα δούμε παρακάτω τα οποία πρέπει να διευθετηθούν.

- **Ρυθμιστικά προβλήματα** : Είναι σημαντικό τα τοκεν να ρυθμίζονται έτσι ώστε οι επενδυτές να νιώθουν σιγουριά, όπως όσο και όταν επενδύουν χρησιμοποιώντας παραδοσιακές χρηματοοικονομικές εταιρίες. Το ζήτημα με την τοκενοποίηση και γενικότερα με το blockchain είναι ότι είναι αποκετρνικοποιημένα και σε περιπτώσεις που καμία οργάνωση δεν έχει τον έλεγχο, είναι δύσκολο να αποδοθούν ευθύνες σε περίπτωση που κάτι πάει στραβά. Σε ένα παραδοσιακό σύστημα, είναι δυνατό να απευθυνθεί κανείς σε ρυθμιστικές αρχές, αλλά στο blockchain ποιος μπορεί να θεωρηθεί υπεύθυνος; Αυτό το πρόβλημα έχει βελτιωθεί με την εφαρμογή προτύπων αξιόγραφων τοκεν και νομοθεσιών, τα οποία θεωρούν τα τοκεν ως αξιόγραφα. Αυτό σημαίνει ότι τα τοκεν θα αντιμετωπίζονται σαν αξιόγραφα, και θα έχουν τις ίδιες νομικές, οικονομικές και κανονιστικές επιπτώσεις που εφαρμόζονται στην ισχύουσα οικονομική βιομηχανία. Αυτό βοηθά στην ενίσχυση της σιγουριάς των επενδυτών όμως υπάρχουν ακόμα πολλές προκλήσεις που πρέπει να διευθετηθούν.[6]
- **Νομιμότητα** : Σε κάποιες δικαιοδοσίες τα τοκεν και τα κρυπτονομίσματα είναι παράνομα.[6]
- **Τεχνολογικό φράγμα** : Τα παραδοσιακά οικονομικά συστήματα χρησιμοποιούνται εδώ και πολύ καιρό και η κοινωνία τα έχει συνηθίσει, όμως τα πράγματα αλλάζουν και με την τοκενοποίηση αυτή η αλλαγή είναι ραγδαία, αυτό έχει σαν αποτέλεσμα, αν και να είναι για πολλούς ανθρώπους ευκολότερο στην χρήση, να είναι πρόβλημα σε τεχνολογικά αναλφάβητους, όπως οι ηλικιωμένοι.[6]
- **Ζητήματα ασφαλείας** : Παρόλο που, όπως είχαμε πει σε προηγούμενο κεφάλαιο, πως οι επιθέσεις στο blockchain είναι πολύ δύσκολες, δεν ισχύει το ίδιο για τις πλατφόρμες

τοκενοοποίησης όπως τα έξυπνα συμβόλαια, όπως αναφέρθηκε σε προηγούμενο κεφάλαιο, οι χακερ μπορούν να εκμεταλλευτούν κενά ασφαλείας στον κώδικα με αποτέλεσμα οικονομικές απώλειες.[6]

3.5 Gas

Όπως είπαμε το δίκτυο του Ethereum λόγο του EVM είναι Turing πλήρες και μπορεί να τρέξει οποιοδήποτε κώδικα. Αυτή η δυνατότητα έρχεται μαζί με ένα μεγάλο αντίτιμο, κάποια προγράμματα μπορεί να χρειαστούν άπειρο χρόνο για να ολοκληρώσουν την εκτέλεση του. Αυτό ονομάζεται πρόβλημα διακοπής (halting problem) και θα ήταν ένα μεγάλο πρόβλημα για το Ethereum αν δεν επιλυόταν. Αυτό συμβαίνει επειδή το EVM λειτουργεί όπως είπαμε σειριακά (σαν έναν μονοπύρηνο επεξεργαστή) έτσι εάν έτρεχε ένα ατέρμονο πρόγραμμα, είτε από κάποιο αθώο λάθος ή από κακή πρόθεση, θα κολλούσε σε έναν ατέρμονο βρόγχο καθιστώντας το τελικά άχρηστο.[3]

Η λύση σε αυτό το πρόβλημα έρχεται με το gas, το οποίο είναι μία μονάδα μέτρηση του Ethereum για την μέτρηση των υπολογιστικών πόρων και τον αποθηκευτικό χώρο που χρειάζεται για την εκτέλεση ενός προγράμματος, αν μετά από μία προκαθορισμένη ποσότητα υπολογιστικών βημάτων δεν έχει τελειώσει η εκτέλεση του προγράμματος, τότε αυτή διακόπτεται από το EVM, κάνοντας το όπως είπαμε σχεδόν Turing πλήρες, ικανό να τρέξει οποιοδήποτε πρόγραμμα με την προϋπόθεση ότι θα τερματίσει η εκτέλεση του μετά από έναν συγκεκριμένο αριθμό υπολογιστικών βημάτων. Αυτό το όριο καθορίζεται μέσω συναίνεσης από όλους που συμμετέχουν στο δίκτυο.[3]

Το gas επίσης δίνεται σαν ανταμοιβή στους miners για την δουλεία που κάνουν. Ο χρήστης που καλεί το συμβόλαιο έχει την δυνατότητα να επιλέξει πόσα ether θέλει να διαθέσει για gas και οι miners επιλέγουν να προσθέσουν, στο μπλοκ που φτιάχνουν, τις συναλλαγές που έχουν το περισσότερο gas.[3]

4 ΚΕΦΑΛΑΙΟ 4ο : Υλοποίηση ενός DApp

Σε αυτό το κεφάλαιο θα αναφέρουμε συνοπτικά τι είναι τα dapp, τα εργαλεία που χρησιμοποιήθηκαν, όπως το metamask και το truffle suite καθώς και μια σύντομη αναφορά σε εργαλεία ανάπτυξης ιστοσελίδων. Επίσης θα αναλυθεί τι είναι ένα dapp, η μεθοδολογία για την ανάπτυξη του dapp καθώς και τα βήματα που ακολουθήθηκαν για την προσθήκη έξυπνων συμβολαίων τα οποία προσθέτουν λειτουργικότητα στην εφαρμογή μας.

4.1 Τι είναι τα DApp?

Τα DApp είναι διαδικτυακές εφαρμογές που είναι μερικώς ή τελείως αποκεντρικοποιημένες. Τα στοιχεία μίας εφαρμογής που μπορούν να αποκεντρικοποιηθούν είναι το backend λογισμικό, το frontend λογισμικό, ο αποθηκευτικός χώρος δεδομένων, η επικοινωνία μηνυμάτων και η επίλυση ονόματος (name resolution). Στα dapps, η αποκεντρικοποίηση του backend γίνεται με την χρήση των έξυπνων συμβολαίων όπου βρίσκεται αποθηκευμένη η λογική της εφαρμογής και οι καταστάσεις της.[26]

Μερικά από τα πιο διαδεδομένα DApp είναι τα Cryptokitties, όπου ο καθένας μπορεί να αγοράσει, να πουλήσει και να δημιουργήσει crypto collectible γάτες και το Idex η οποία είναι μία αποκεντρικοποιημένη υπηρεσία συναλλάγματος για crypto νομίσματα.

4.2 Ανάπτυξη αποκεντρικοποιημένων εφαρμογών

Παρακάτω θα δούμε τα εργαλεία που χρειάστηκαν για την υλοποίηση της αποκεντρικοποιημένης εφαρμογής όπως οι γλώσσες προγραμματισμού για την ανάπτυξη ιστοσελίδων HTML, CSS και JavaScript για το front end, τα εργαλεία για την δημιουργία ενός τοπικού blockchain όπως το ganache και εργαλεία για την ανάπτυξη της εφαρμογής όπως η γλώσσα solidity.

4.2.1 Εργαλεία ανάπτυξης ιστοσελίδας

Κάθε DApp χρειάζεται ένα user interface(UI) για να μπορούν οι χρήστες να αλληλοεπιδρούν με αυτήν. Για τις ανάγκες της διπλωματικής υλοποιήθηκε μία απλή ιστοσελίδα.

4.2.2 HTML

Η HTML, ή αλλιώς HyperText Markup Language, είναι μία γλώσσα σήμανσης η οποία ορίζει την δομή της ιστοσελίδας. Η HTML αποτελείται από στοιχεία τα οποία «τυλίγουν» διάφορα τμήματά των περιεχόμενών έτσι ώστε να εμφανίζονται η να ενεργούν με κάποιον συγκεκριμένο τρόπο.[13]

4.2.3 CSS

Η CSS, ή αλλιώς Cascading Style Sheet, είναι μία γλώσσα με την οποία μπορούμε επιλεκτικά να στολίσουμε τα στοιχεία της HTML έτσι ώστε να βελτιώσουμε την εμφάνιση της ιστοσελίδας.[13]

4.2.4 Javascript

Η JavaScript είναι μία πλήρως δυναμική γλώσσα προγραμματισμού, η οποία μπορεί να προσθέσει δια δραστικότητα σε μία ιστοσελίδα. Για τις ανάγκες της διπλωματικής χρησιμοποιήθηκε η JavaScript βιβλιοθήκη jQuery.[13]

4.3 Εργαλεία blockchain

Σε αυτήν την ενότητα θα αναφερθούμε στα εργαλεία που χρησιμοποιήθηκαν για την ανάπτυξη και δοκιμή του DApp σε ένα τοπικό blockchain το οποίο υλοποιήθηκε τοπικά λόγο θεωρήθηκε γρηγορότερο και ευκολότερο στην χρήση για την ανάπτυξη και την αποσφαλμάτωση της αποκεντρικοποιημένης εφαρμογής.

4.3.1 NPM

Το NPM (Node Package Manager) είναι ένα λογισμικό για την διαχειριστή και την εγκατάσταση πακέτων για την πλατφόρμα Node JavaScript.[24]

4.3.2 Truffle suite

Το Truffle suite είναι ένα περιβάλλον για την ανάπτυξη DApps, framework για την δοκιμή των DApp χρησιμοποιώντας το Ethereum Virtual Machine. Είναι βασισμένο στο blockchain του Ethereum και είναι σχεδιασμένο για να διευκολύνει την ανάπτυξη των DApps. Με το truffle μπορούμε να κάνουμε compile και deploy τα smart contracts και να τα συνδέσουμε με διαδικτυακές εφαρμογές.[30]

4.3.3 Ganache

Το ganache είναι ένα τοπικό blockchain, το οποίο χρησιμοποιείται για την ανάπτυξη, δοκιμή και deployment των DApp σε ένα ασφαλή και ντετερμινιστικό περιβάλλον.[30]

4.3.4 Solidity

Η solidity είναι μία υψηλού επιπέδου γλώσσα προγραμματισμού η οποία χρησιμοποιείται για την ανάπτυξη έξυπνων συμβολαίων. Χρησιμοποιείται για την σχεδίασή και εφαρμογή έξυπνων συμβολαίων στο Ethereum Virtual Machine. λόγο ότι είναι σχεδιασμένη βάση του συντακτικού της JavaScript είναι ευκολότερο για τους web developers να την καταλάβουν και να την εφαρμόσουν.[34]

4.3.5 Metamask

Το metamask είναι ένα ψηφιακό πορτοφόλι για κρυπτονομίσματα και μία πύλη διασύνδεσης σε εφαρμογές στο blockchain.[20] Θα το χρησιμοποιήσουμε για να συνδεθούμε στο τοπικό blockchain που θα μας δημιουργήσει το ganache καθώς και για να κάνουμε συναλλαγές στο DApp που θα υλοποιήσουμε.

4.3.6 Remix

Το remix είναι ένα solidity IDE (Integrated Development Environment) το οποίο χρησιμοποιείται για την συγγραφή, το compile και την αποσφαλμάτωση κώδικα solidity.[9]

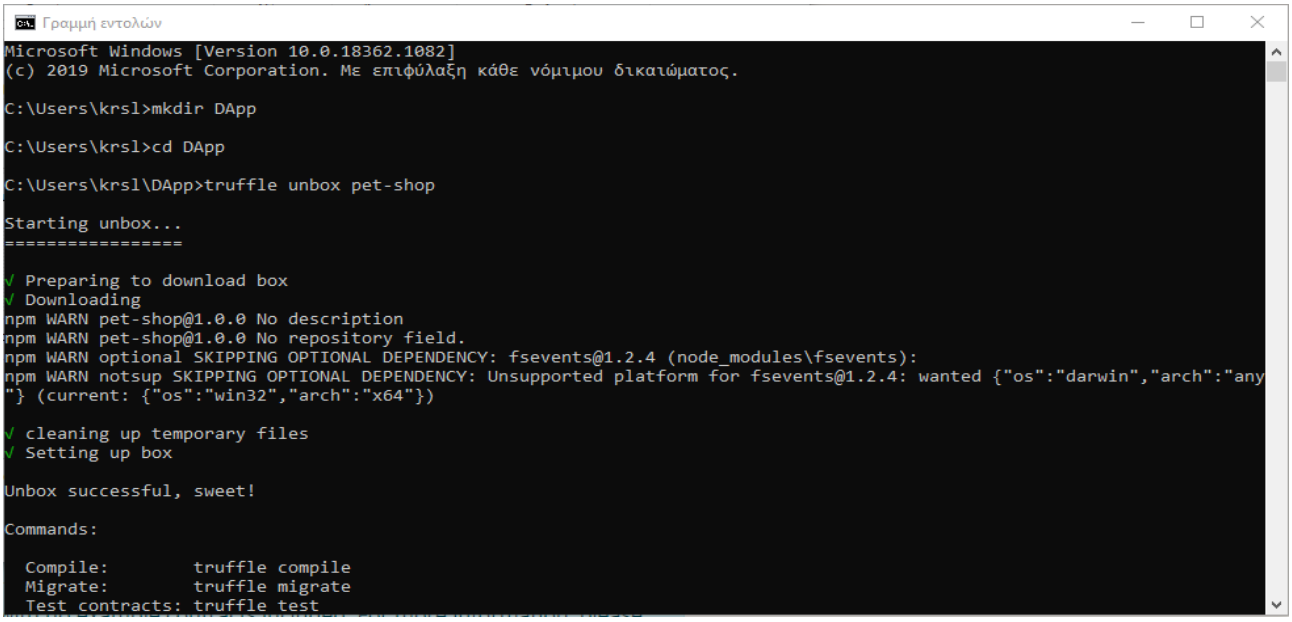
4.4 Η υλοποίηση του DApp μας

Για την υλοποίηση του DApp, αρχικά γράψαμε τις συναρτήσεις του έξυπνου συμβολαίου κάνοντας για κάθε συνάρτηση κάθε φορά την αποσφαλμάτωση με την χρήση της ιστοσελίδας Remix για να σιγουρευτούμε ότι ο κώδικας μας τρέχει σωστά και χωρίς προβλήματα και αντιγράψαμε τον κώδικα για το ERC721 standard από την ιστοσελίδα http://erc721.org/ σε ένα άλλο contract. Αφού σιγουρευτούμε ότι ο κώδικας λειτουργεί όπως πρέπει προχωρήσαμε στην υλοποίηση της ιστοσελίδας με την οποία θα γίνει η διασύνδεση του έξυπνου συμβολαίου με το front end καθώς και το UI για την αλληλεπίδραση του χρήστη με την εφαρμογή μας. Μόλις τελειώσουμε και με αυτό το κομμάτι του κώδικα δημιουργήσαμε με το ganache και το npm ένα τοπικό blockchain και ένα τοπικό δίκτυο αντίστοιχα και παίξαμε με το DApp μας εξαντλώντας όλες τις περιπτώσεις για να σιγουρευτούμε ότι το front end λειτουργεί επίσης σωστά. Με την ίδια λογική υλοποιήσαμε ένα δεύτερο DApp για να δείξουμε την διασύνδεση μεταξύ δύο DApp, με την προϋπόθεση ότι υπάρχει ο κατάλληλος κώδικας στο κύριο DApp.

Το DApp που υλοποιήθηκε είναι ένα text based game, χωρίς κάποια συγκεκριμένη υπόθεση, στο οποίο κάθε χρήστης μπορεί να φτιάξει έναν μόνο χαρακτήρα πληρώνοντας το ανάλογο gas. Με αυτόν το χαρακτήρα οι χρήστες μπορούν να «περπατάνε» και κάθε φορά που κάνουν ένα βήμα

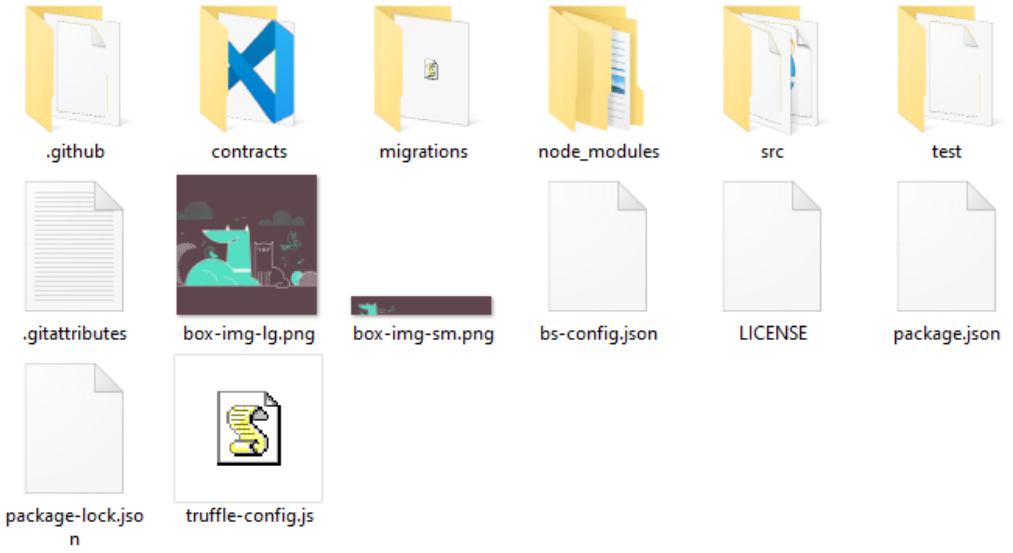
υπάρχει η πιθανότητα εμφάνισης εχθρού με στον οποίο μπορούν είτε να επιτίθενται μέχρι ένας από τους δύο να πεθάνει, είτε οποιαδήποτε στιγμή να τρέξει από την μάχη αποφεύγοντας τον εχθρό. Στην περίπτωση που πεθάνει ο χαρακτήρας τελειώνει η μάχη και επαναφέρεται η ζωή του χαρακτήρα στο μέγιστο. Στην περίπτωση που νικήσει υπάρχει μία πιθανότητα να πέσει ένα crypto collectible το οποίο θα προστεθεί στην συλλογή του χρήστη. Αυτό το crypto collectible, θα έχει την δυνατότητα να το πουλήσει ο χρήστης στην τιμή που επιθυμεί ή αντίστοιχα να αγοράσει crypto collectibles από άλλους χρήστες. Όσον αφορά το δεύτερο DApp αποτελείται από μία φόρμα στην οποία οι χρήστες μπορούν να αναζητήσουν βάση του id χαρακτήρες από το κύριο DApp.

Αρχικά κάνουμε εγκατάσταση του truffle suite, χρησιμοποιώντας τις εξής εντολές στην γραμμή εντολών. Mkdir DApp, για να φτιάξουμε στον υπολογιστή μας έναν φάκελο με το όνομα DApp ο οποίος θα φιλοξενήσει την εφαρμογή μας. cd DApp για να αλλάξουμε στην γραμμή εντολών, σε αυτήν του DApp. Και truffle unbox pet-shop με την οποία κατεβάζουμε ένα truffle box με ένα DApp παράδειγμα το οποίο περιέχει ότι αρχεία θα χρειαστούμε για την ανάπτυξη του DApp μας (εικόνα 4.1).



Εικόνα 4.1 Εγκατάσταση πακέτων του truffle

Αυτό θα μας δώσει τα παρακάτω αρχεία της εικόνας 4.2 από τα οποία θα διαγράψουμε ότι δεν χρειαζόμαστε.



Εικόνα 4.2 Αρχεία που δημιουργήθηκαν από την εγκατάσταση του truffle

Αρχικά θα επεξεργαστούμε το αρχείο truffle-config.js έτσι ώστε να μπορέσουμε να τρέξουμε το DApp μας σε ένα τοπικό δίκτυο, τροποποιώντας τον κώδικα του αρχείου σε αυτόν που βλέπουμε στην εικόνα 4.3.

```
module.exports = {
  // See <http://truffleframework.com/docs/advanced/configuration>
  // for more about customizing your Truffle configuration!
  networks: {
    development: {
      host: "127.0.0.1",
      port: 8545,
      network_id: "*" // Match any network id
    }
  },
  compilers: {
    solc: {
      version: '0.4.25',
      optimizer: {
        enabled: true,
        runs: 200
      }
    }
  }
};

```

Εικόνα 4.3 Τροποποιημένος κώδικας του αρχείου truffle-config.js

Στην συνέχεια θα πάμε στον φάκελο migrations και θα μετονομάσουμε το αρχείο σε 2_deploy_contracts.js και τροποποιώντας τον κώδικα όπως φαίνεται παρακάτω (εικόνα 4.4). Αυτό το αρχείο θα μας κάνει deploy το έξυπνο συμβόλαιο μας στο τοπικό Ethereum blockchain.

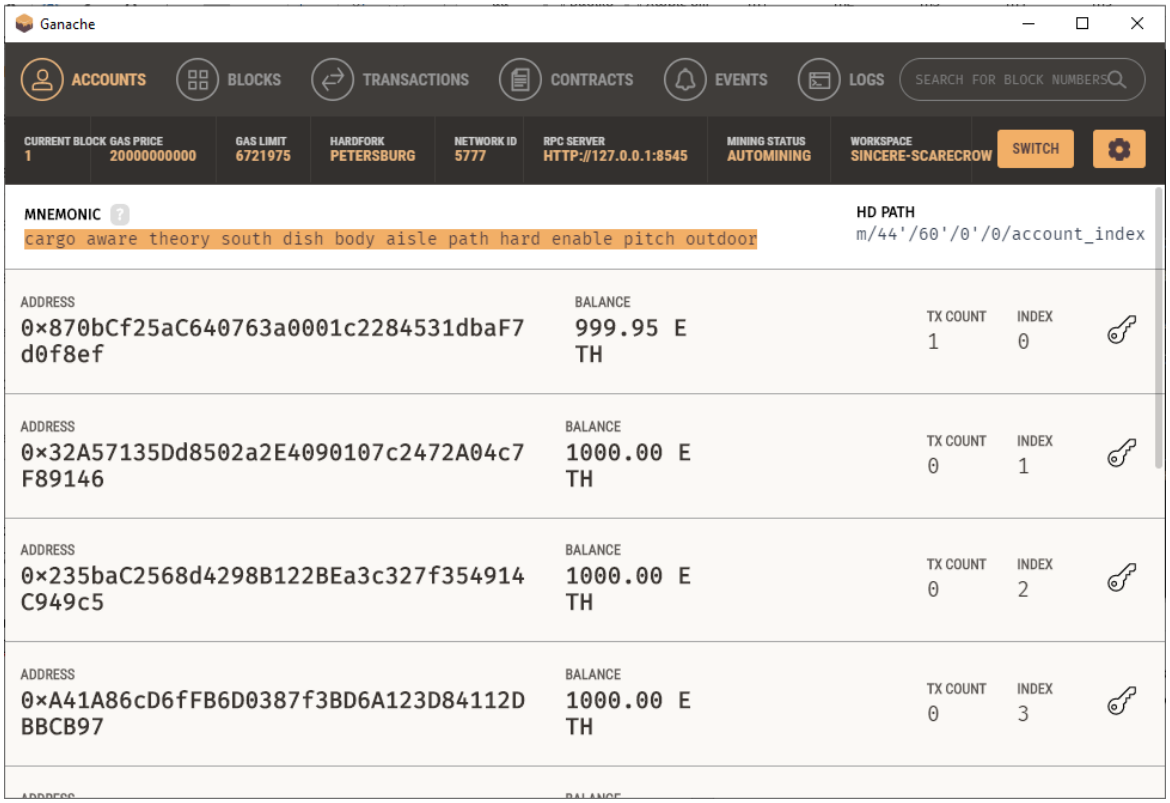
```
1  var Creation = artifacts.require("./Creation.sol");
2
3  module.exports = function(deployer) {
4
5      deployer.deploy(Creation);
6
7  };
```

Εικόνα 4.4 Τροποποίηση κώδικα του αρχείου 2_deploy_contracts.js

Προχωράμε με την υλοποίηση του DApp μας. Πάμε στον φάκελο contracts και αυθαίρετα μετονομάζουμε το αρχείο Migrations σε Creation, το ανοίγουμε, και αφότου διαγράψουμε τα περιεχόμενα του πλην του σκελετού και ξεκινάμε την συγγραφή των smart contracts τα οποία θα δούμε αργότερα κατά την παρουσίαση του DApp.

Όπως αναφέρθηκε σε προηγούμενη υπό ενότητα κάθε DApp χρειάζεται ένα UI και το truffle box περιέχει τα απαραίτητα αρχεία από τα οποία θα διαγράψουμε τα περιεχόμενα τους και θα τα προγραμματίσουμε κατάλληλα έτσι ώστε να λειτουργήσει το DApp μας.

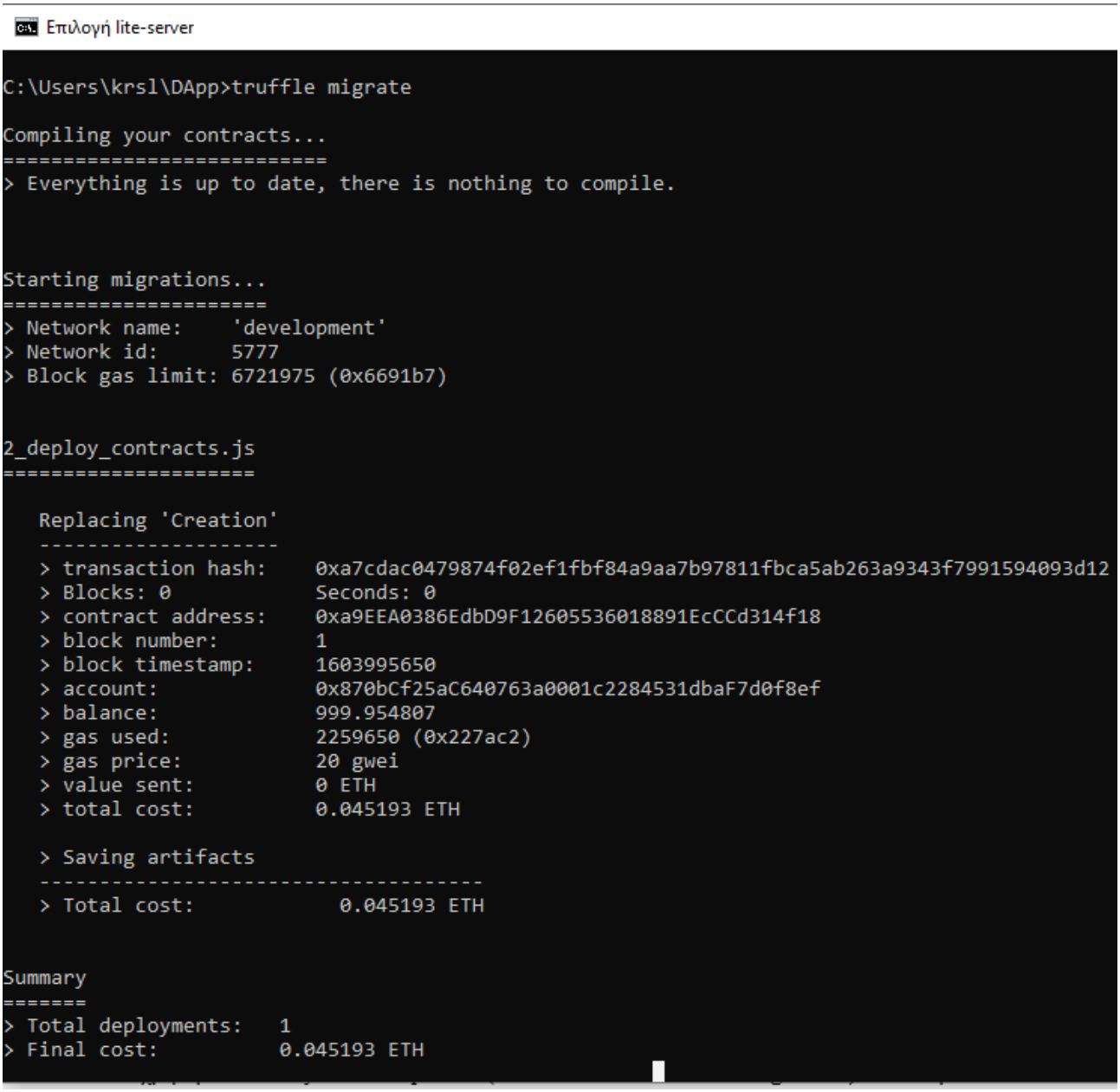
Έπειτα με την χρήση του ganache δημιουργούμε ένα τοπικό blockchain, θέτοντας την ρύθμιση port number σε 8545, το οποίο μας δίνει δέκα λογαριασμούς με 100 ETH ο καθένας. Στην εικόνα 4.5 βλέπουμε το UI του ganache.



Εικόνα 4.5 UI του ganache

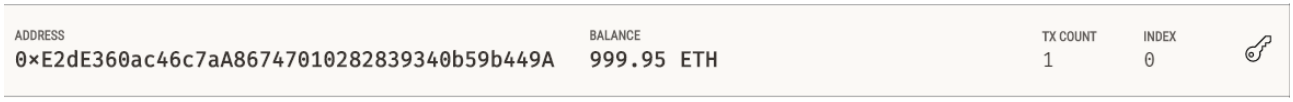
Μετά από αυτό, επιλέγοντας στο Metamask το δίκτυο τοπικός υπολογιστής 8545 και χρησιμοποιώντας το seed phrase (MNEMONIC στο UI του ganache) συνδεόμαστε το τοπικό blockchain που δημιούργησε το ganache.

Αφού τα κάνουμε όλα αυτά, τρέχουμε την εντολή truffle deploy στην γραμμή εντολών για να ανεβάσουμε το DApp μας στο τοπικό δίκτυο blockchain και την εντολή npm run dev για να δημιουργήσουμε ένα τοπικό δίκτυο για την ιστοσελίδα του DApp (εικόνα 4.6).



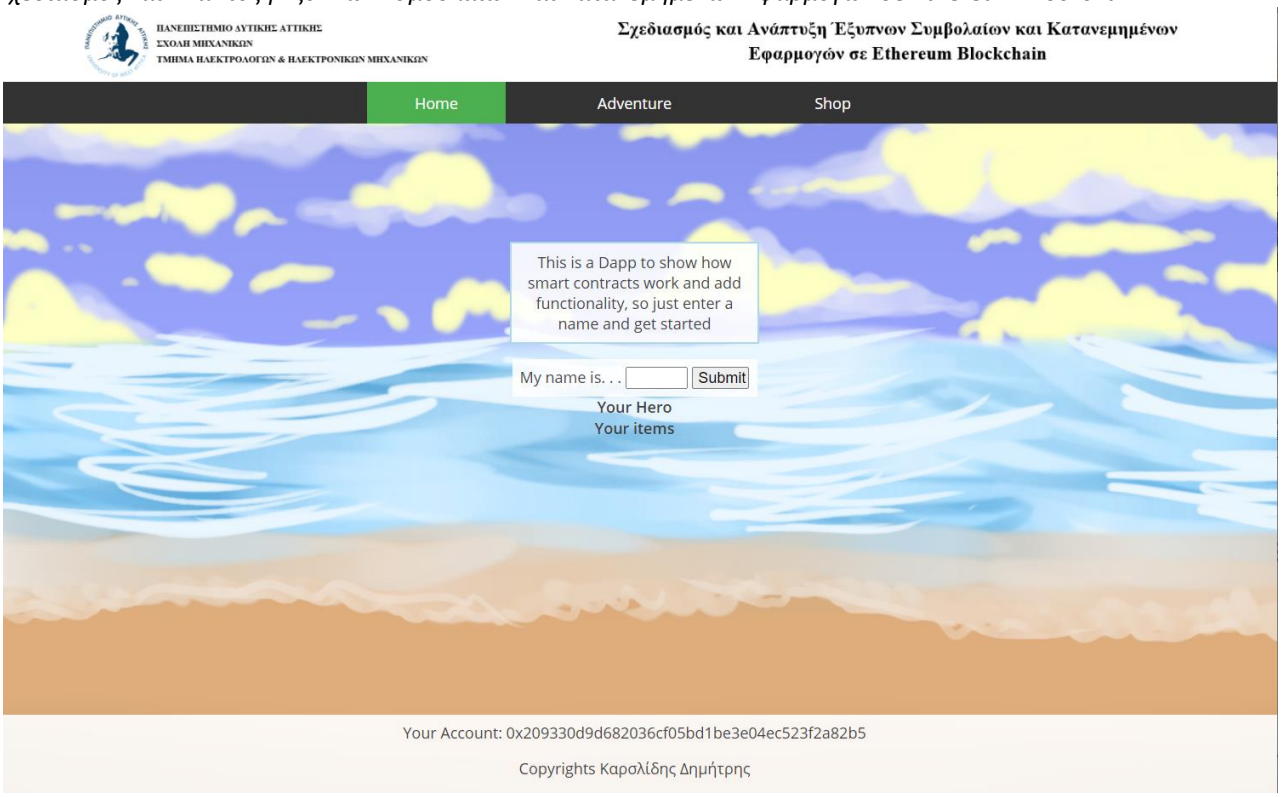
Εικόνα 4.6 Contract deployment στο Ethereum Blockchain

Όπως γνωρίζουμε για να γίνει εγγραφή στο blockchain πρέπει να πληρώσουμε gas, έτσι και κατά το deployment του DApp έχουμε το αντίστοιχο κόστος το οποίο αφαιρέθηκε από το πρώτο λογαριασμό του ganache (εικόνα 4.7).



Εικόνα 4.7 υπόλοιπο στο λογαριασμό μετά από deployment

Επίσης τρέχουμε την εντολή npm run dev για να φτιάξουμε ένα τοπικό δίκτυο στο οποίο θα τρέξουμε το DApp. παρακάτω μπορούμε να δούμε την ιστοσελίδα για το Dapp (εικόνα 4.8).



Εικόνα 4.8 Ιστοσελίδα του DApp

4.5 Παρουσίαση του DApp και των συναρτήσεων του έξυπνου συμβολαίου

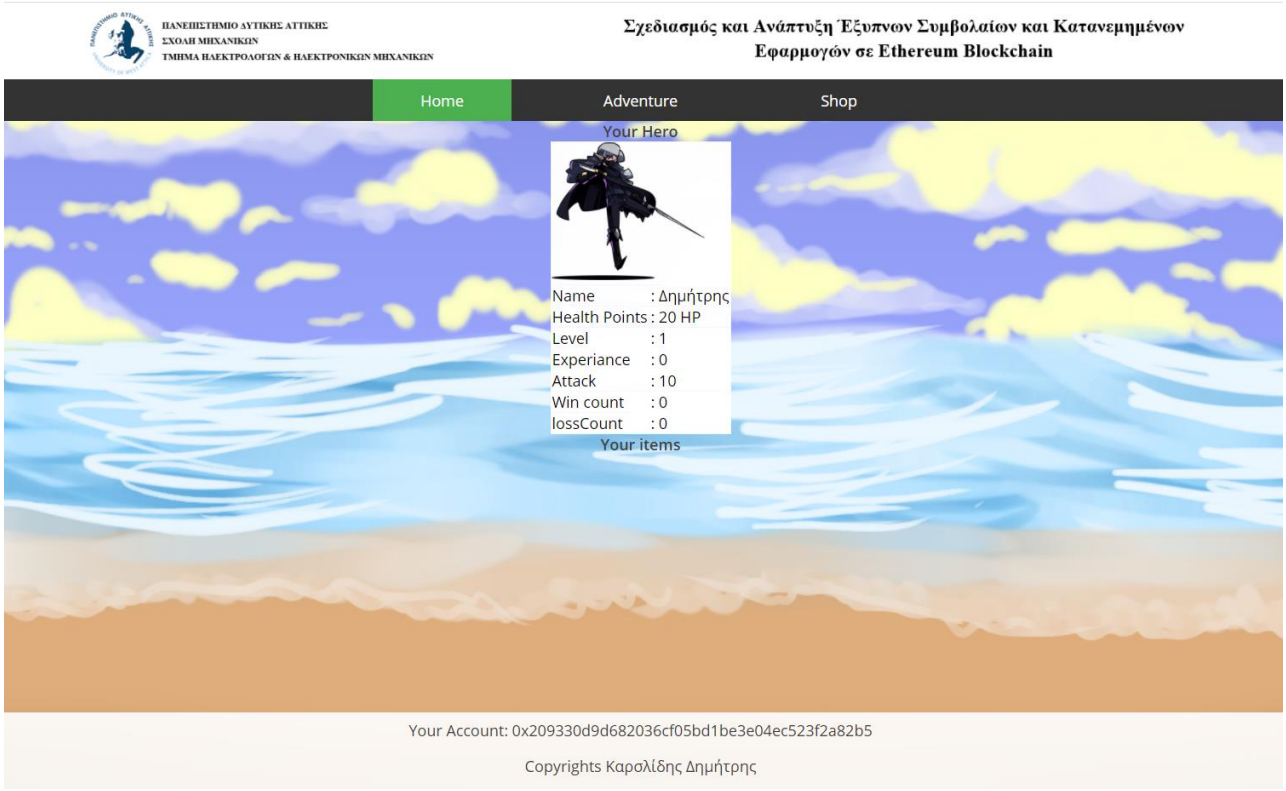
Αφού ολοκληρώσουμε την παραπάνω διαδικασία μπορούμε να ξεκινήσουμε την παρουσίαση του DApp. Αρχικά πρέπει να φτιάξουμε έναν χαρακτήρα εισάγοντας ένα όνομα στο αντίστοιχο πεδίο και να δούμε την πρώτη λειτουργικότητα του DApp μέσο του έξυπνου συμβολαίου. Μόλις κάνουμε submit θα γίνει κλήση της συνάντησης createhero (εικόνα 4.9) του έξυπνου συμβολαίου μας, η οποία μας δημιουργεί τον χαρακτήρα μας, με την προϋπόθεση ότι έχουμε πληρώσει το gas.

Πιο αναλυτικά, η συνάρτηση μας παίρνει σαν παράμετρό ένα string, το οποίο το επιλέγουμε εμείς μέσω της φόρμας του UI. Αυτή η συνάρτηση για να κληθεί χρειάζεται ο χρήστης να μην έχει άλλο χαρακτήρα. Στην συνέχεια, προσθέτει τον χαρακτήρα, με τα διάφορα στατιστικά του, όπως το όνομα και την ζωή σε έναν πίνακα με όλους τους ήρωες και τοποθετεί την τιμή σε μια μεταβλητή id. Μετά από αυτό βάση του id τοποθετούμε την διεύθυνση του χρήστη σε έναν πίνακα, ο οποίος συνδέει τον χαρακτήρα και τον ιδιοκτήτη του. Τέλος αυξάνεται ο αριθμός των χαρακτήρων που έχει ο χρήστης έτσι ώστε να μην μπορεί να φτιάξει άλλον χαρακτήρα.

```
48    function createhero(string memory _name) public {
49      require(heroCount[msg.sender] == 0);
50        uint id= heroes.push(hero(_name,basiclife,1,0,15,0,0,0))-1;
51        herotoowner[id] = msg.sender;
52        heroCount[msg.sender]++;
53        emit NewHero(id, _name);
54    }
```

Εικόνα 4.9 Συνάρτηση έξυπνου συμβολαίου για την δημιουργία χαρακτήρα

Το αποτέλεσμα η δημιουργία του χαρακτήρα μας. (εικόνα 4.10)



Εικόνα 4.10 Ο χαρακτήρας που δημιουργήθηκε από το έξυπνο συμβόλαιο

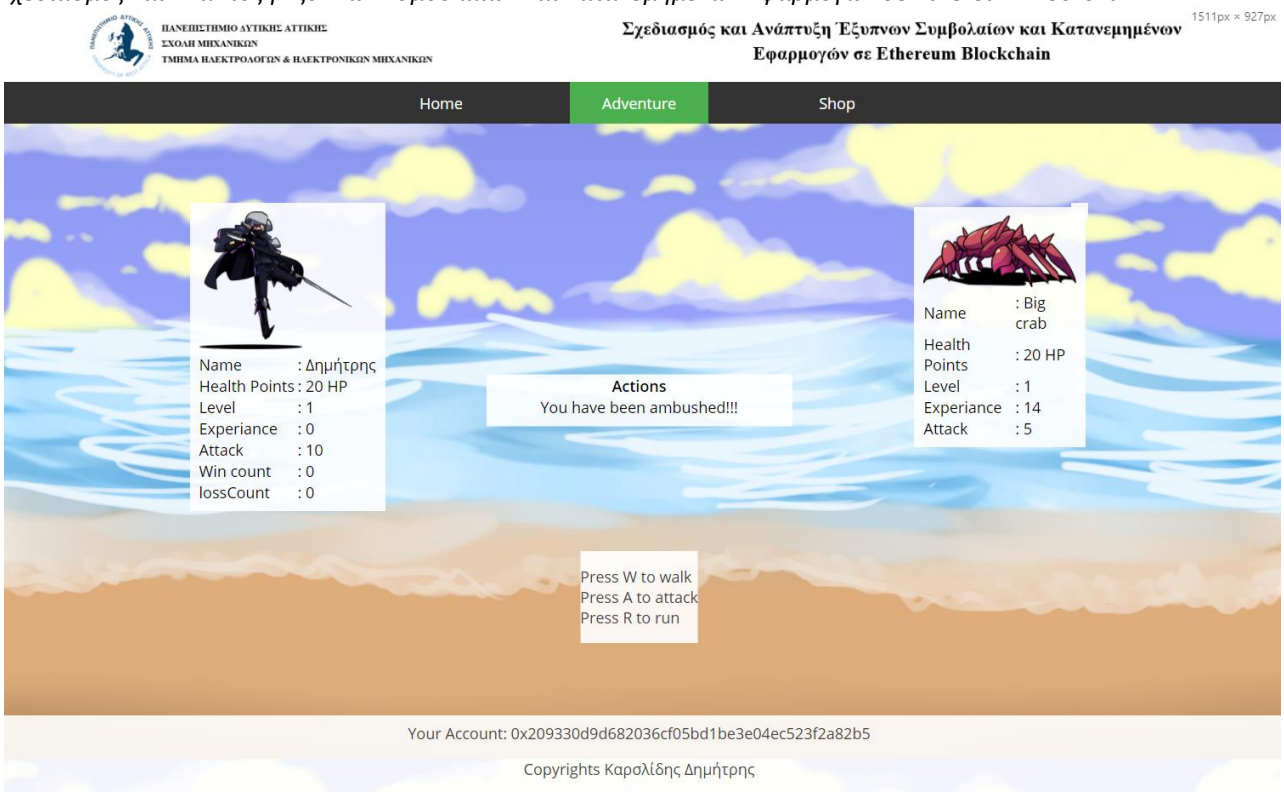
Συνεχίζουμε με την επόμενη συνάρτηση του έξυπνου συμβολαίου πηγαίνοντας στην σελίδα Adventure,εκεί έχουμε την επιλογή να ξεκινήσουμε την περιπέτεια μας, κάθε φορά που ο χαρακτήρας μας περπατάει έχει την πιθανότητα να κληθεί η συνάρτηση που μας δημιουργεί έναν εχθρό όταν την καλέσουμε. Για το DApp μας υλοποιήθηκαν δύο συναρτήσεις, μία για κάθε διαφορετικό τέρας (εικόνα 4.11).

Παρομοίως, όπως και στην δημιουργία του χαρακτήρα με μερικές διαφορές, δημιουργείται και το τέρας. Η συνάρτηση παίρνει σαν παράμετρο το επίπεδο του χαρακτήρα έτσι ώστε να έχει το τέρας ανάλογα στατιστικά. Δημιουργείτε ένα τέρας με τα στατιστικά του και το τοποθετεί σε μία μεταβλητή id, όπου βάση αυτής τοποθετούμε την διεύθυνση του χρήστη που δημιούργησε το τέρας σε ένα πίνακα, ο οποίος συνδέει το τέρας με αυτόν που το δημιούργησε

```
55        function enemyScrab(int _herolvl) public  {
56            uint id=enemies.push(enemy("Small crab",15*_herolvl,_herolvl,7*_herolvl,5*(_herolvl),0))-1;
57            monstertoowner[id] = msg.sender;
58            monsterCount++;
59        }
60        function enemyBcrab(int _herolvl) public  {
61            uint id=enemies.push(enemy("Big crab",20*_herolvl,_herolvl,14*_herolvl,5*(_herolvl),0))-1;
62            monstertoowner[id] = msg.sender;
63            monsterCount++;
```

Εικόνα 4.11 Συνάρτηση έξυπνου συμβολαίου για την δημιουργία εχθρού

Κατά την κλήση της συνάρτησης δημιουργήθηκε το τέρας της εικόνας 4.12

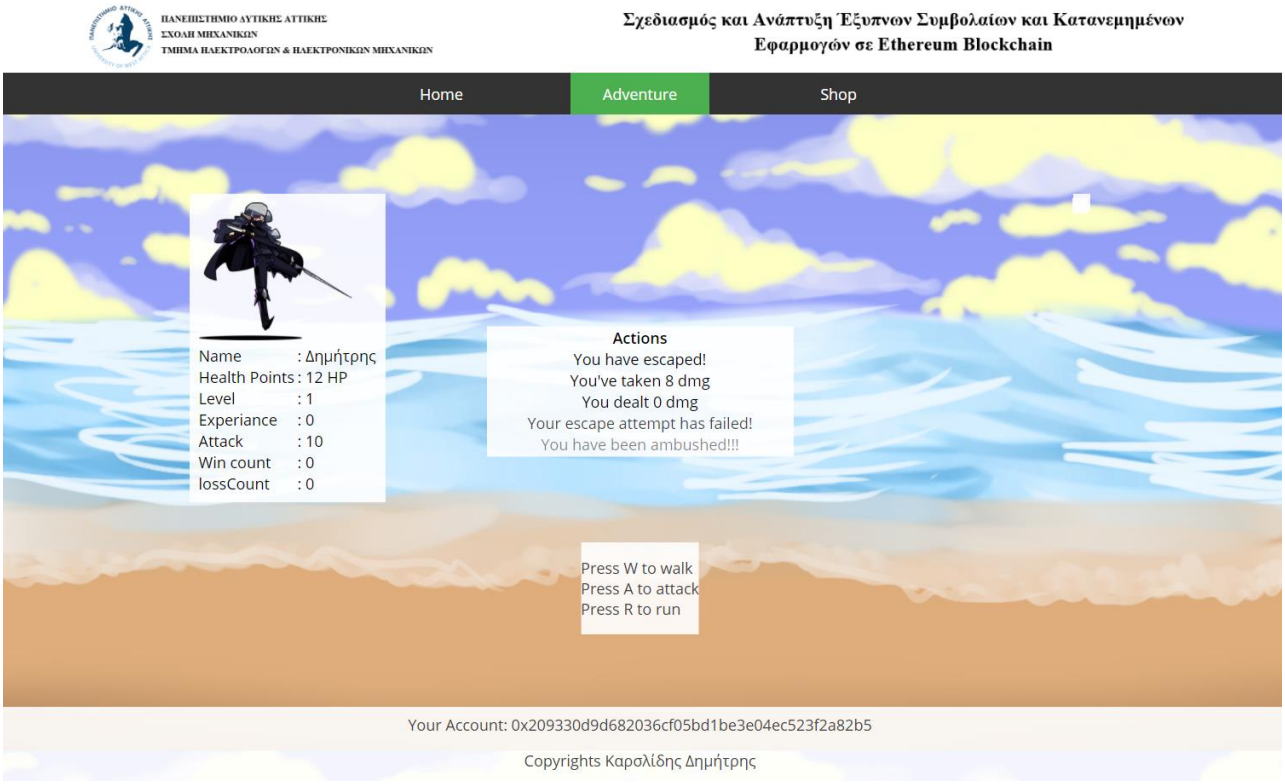


Εικόνα 4.12 Ο εχθρός που δημιουργήθηκε από το έξυπνο συμβόλαιο

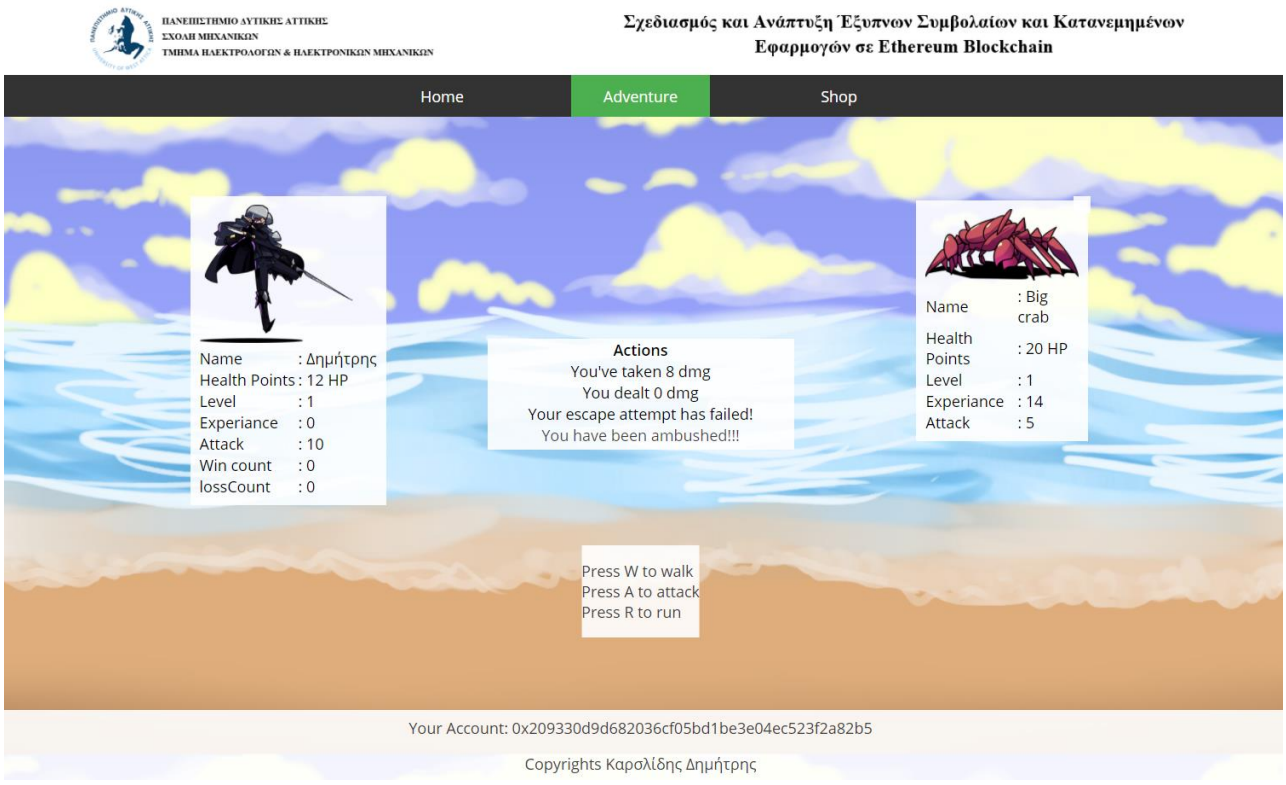
Η επόμενη συνάρτηση του έξυπνου συμβολαίου είναι η επίθεση (εικόνα 4.15), την οποία χρησιμοποιούμε για δύο διαφορετικές λειτουργίες τους παιχνιδιού.

Η πρώτη είναι με την επιλογή run, για να αποφύγουμε την μάχη με το τέρας. Πατώντας το αντίστοιχό πλήκτρο, παράγεται ένας τυχαίος αριθμός είτε μηδέν είτε ένα, όπου αντίστοιχα επιτυγχάνεται ή αποτυγχάνει η αποφυγή. Στην περίπτωση που αποτύχει, η συνάρτηση για την επίθεση παίρνει μία παράμετρο η οποία αν λάβει την τιμή μηδέν πραγματοποιεί ένα γύρο μάχης στην οποία δέχεται ζημιά μόνο ο χαρακτήρας και όχι το τέρας.

Στην εικόνα 4.13 βλέπουμε τον Χαρακτήρα μας όταν πάμε να αποφύγουμε την μάχη πετυχημένα και στην εικόνα 4.14 όταν η αποφυγή αποτυγχάνει.



Εικόνα 4.13 Ο χαρακτήρας μετά από επιτυχημένη αποφυγή της μάχης



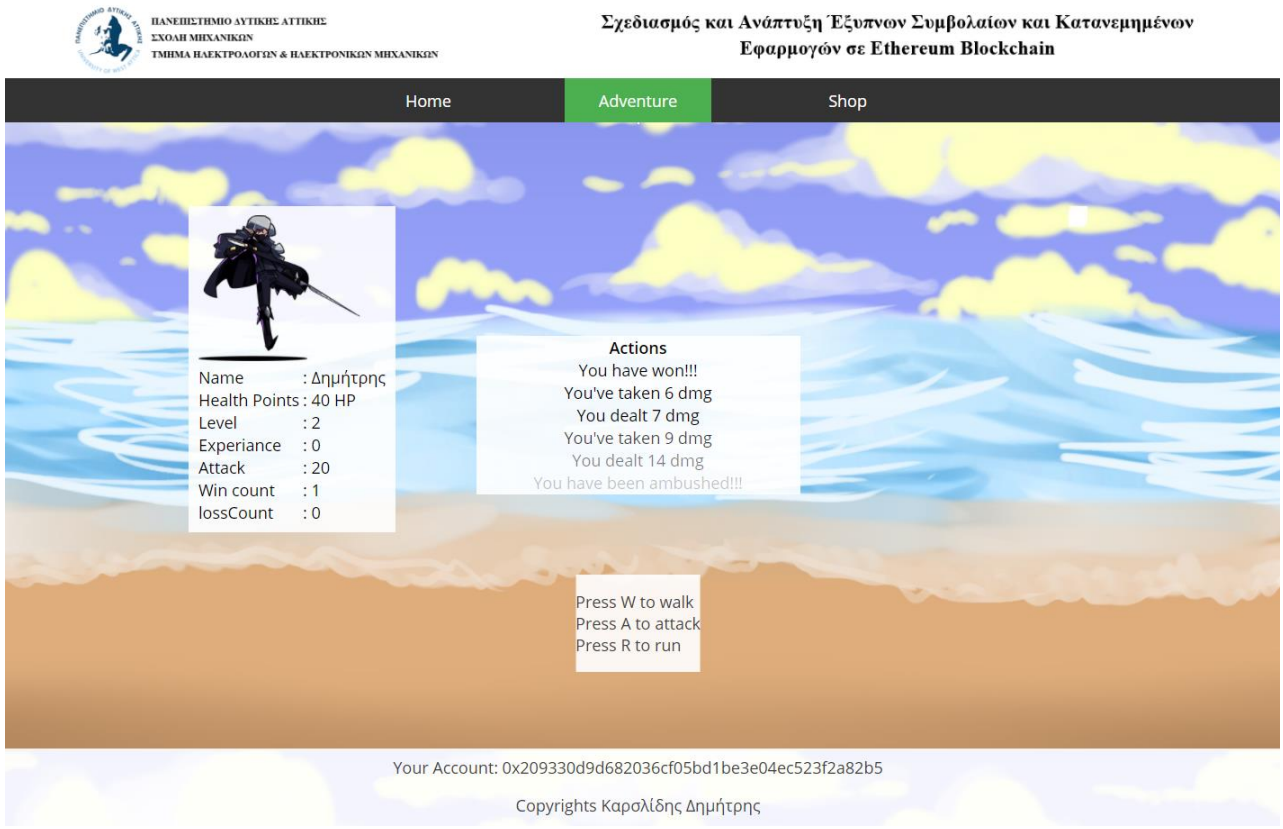
Εικόνα 4.14 Ο χαρακτήρας μετά από αποτυχημένη αποφυγή της μάχης

Η δεύτερη είναι με την επιλογή attack όπου ο χαρακτήρας μας και το τέρας ανταλλάζουν χτυπήματα, προκαλώντας ζημιά ο ένας στον άλλον. Αυτή η ενέργεια μπορεί να επαναληφθεί έως ότου είτε ο χαρακτήρας είτε το τέρας (ή και οι δύο) πεθάνουν.

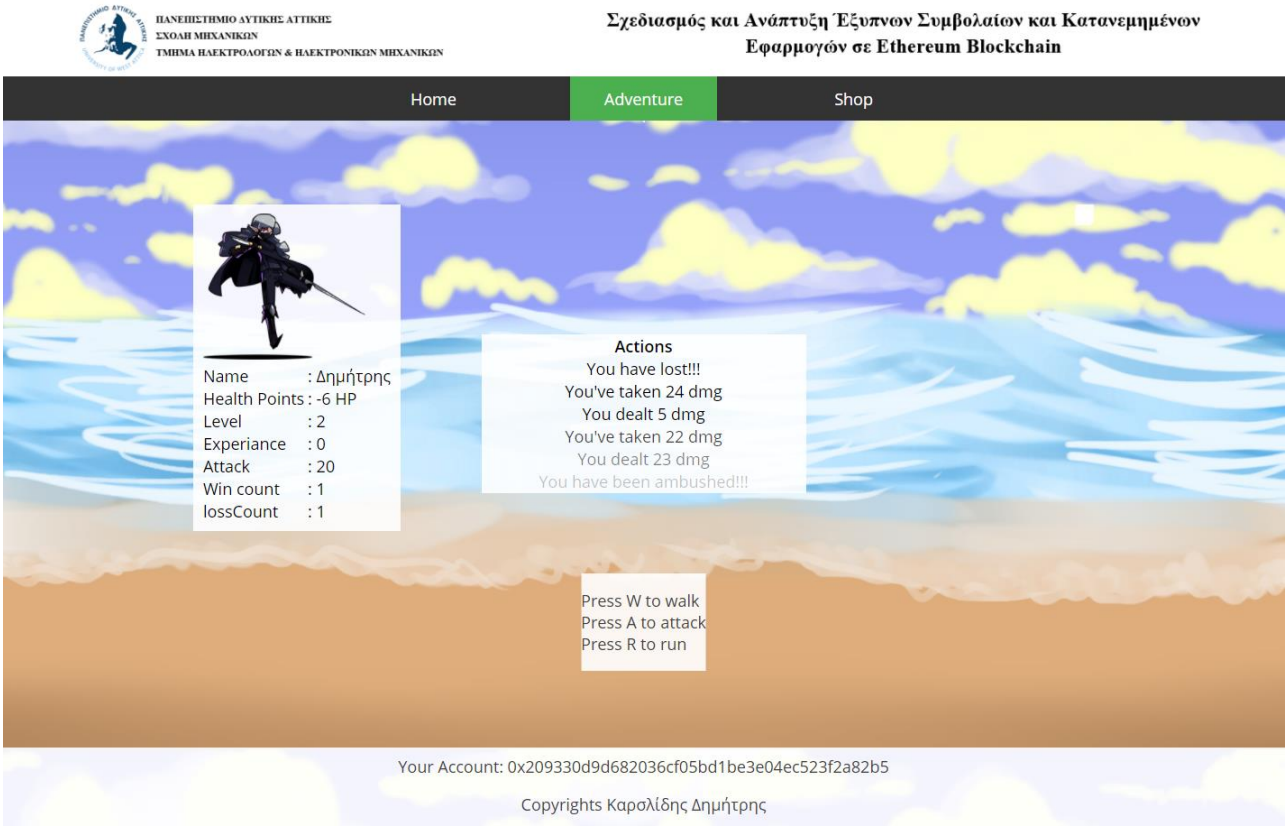
Η συνάρτηση για την επίθεση(εικόνα 4.15), στην περίπτωση της νίκης (εικόνα 4.16), αυξάνει το επίπεδο του χαρακτήρα εάν έχει μαζέψει αρκετούς πόντους εμπειρίας, το οποίο έχει σαν αποτέλεσμα να αυξηθούν και τα στατιστικά του, λαμβάνει πόντους εμπειρίας, αυξάνει ο μετρητής νικών του και έχει πιθανότητα να πάρει ένα crypto collectible. Στην περίπτωση της ήττας (εικόνα 4.17), αυξάνεται ο μετρητής ηττών, εξαφανίζεται το τέρας και επαναφέρεται στην ζωή (εικόνα 4.19) με την συνάρτηση που θα δούμε στην συνέχεια.

```
178      function attack_hero(uint _heroid, uint _monsterid, int _num  ) external  {
179
180        hero storage myhero = heroes[_heroid];
181        enemy storage monster = enemies[_monsterid];
182        int attackofhero=0;
183      int attackofmonster=randNum(monster.attack + 5);
184        if(_num==0 ) {
185         attackofhero=randNum(myhero.attack + 5);
186        }
187        if (myhero.life>0){
188          monster.life=monster.life - attackofhero;
189        }
190        if(monster.life>0){
191          myhero.life=myhero.life - attackofmonster;
192        }
193        myhero.herodmg=attackofhero;
194        monster.mondmg=attackofmonster;
195
196        if (myhero.life <= 0 ) {
197         myhero.lossCount++;
198        }
199        if (monster.life <= 0 && myhero.life >= 1) {
200          myhero.winCount++;
201          myhero.xp+=monster.xp;
202          if (randNum(100) > 0 ){
203                createitem();
204          }
205          if (myhero.xp>=10+2*myhero.lvl){
206            myhero.lvl++;
207            myhero.xp=0;
208            myhero.life=basiclife*myhero.lvl;
209            myhero.attack=myhero.attack*myhero.lvl;
210          }
211       }
212      }
```

Εικόνα 4.15 Συνάρτηση για την επίθεση



Εικόνα 4.16 Νίκη του χαρακτήρα

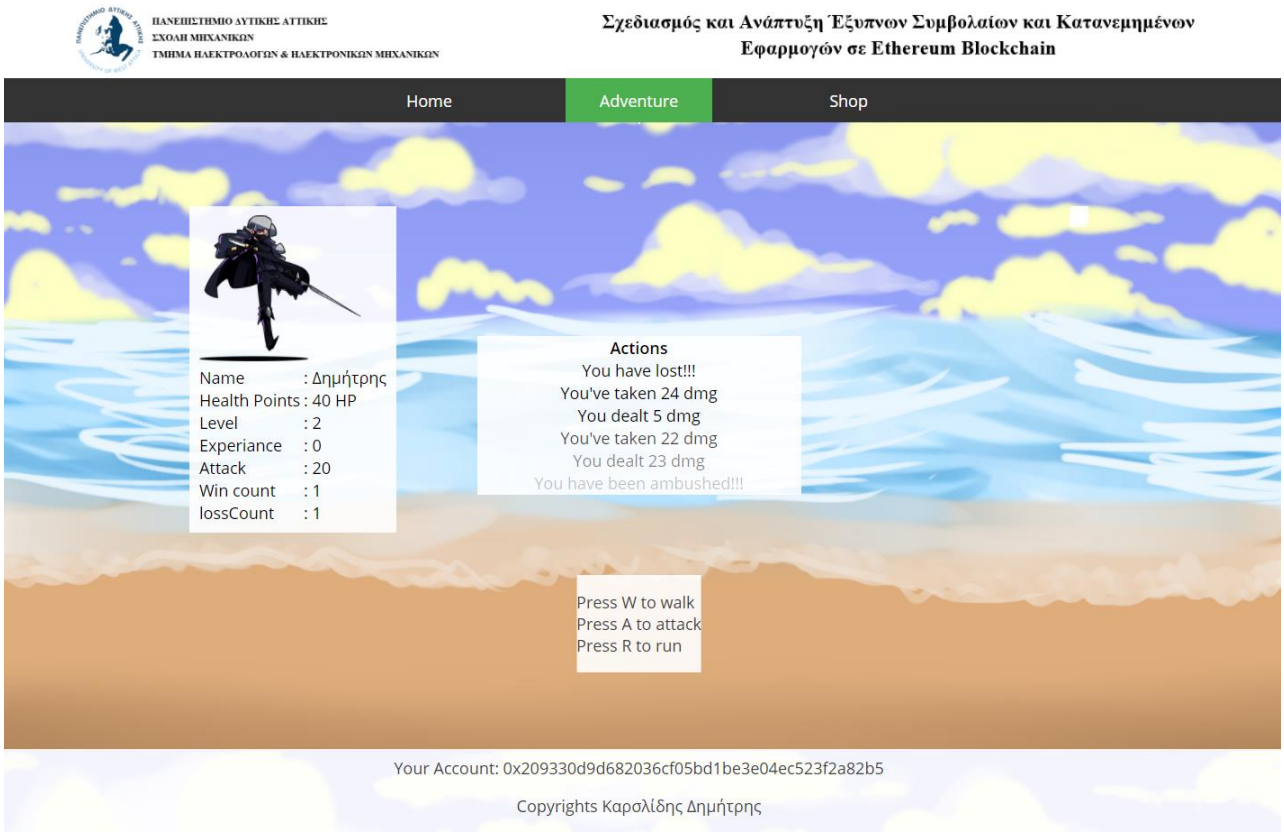


Εικόνα 4.17 ήττα του χαρακτήρα

Προχωράμε στην επόμενη συνάρτηση (εικόνα 4.18), η οποία επιτρέπει στον χαρακτήρα να αναγεννηθεί εάν έχει πεθάνει επαναφέροντας όλους τους πόντους ζωής του χωρίς όμως την δυνατότητα να μπορεί να συνεχίσει την μάχη με το ίδιο τέρας, όπως βλέπουμε στην εικόνα.

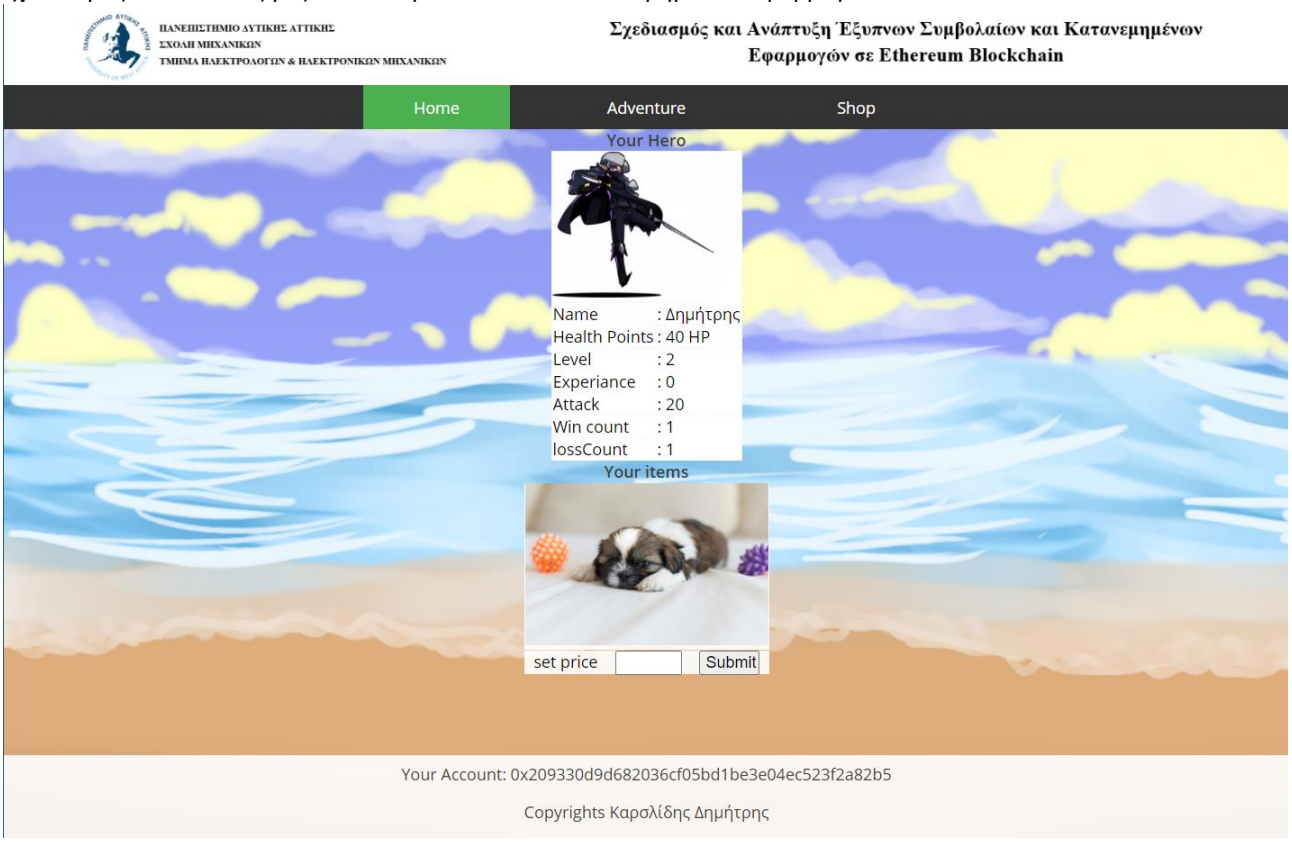
```
213         function resurrection(uint _heroid) public {
214             hero storage myhero = heroes[_heroid];
215             myhero.life=basiclife*myhero.lvl;
216         }
```

Εικόνα 4.18 Συνάρτηση αναγέννησης



Εικόνα 4.19 Ο ήρωας μετά από ήττα

Όπως αναφέραμε, κάθε φορά που νικάει ο ήρωας μας υπάρχει η πιθανότητα να πέσει ένα crypto collectible, επιστρέφοντας στην αρχική σελίδα μπορούμε να το δούμε (εικόνα 4.20) το οποίο δημιουργείται μέσω της συνάρτησης της εικόνας 4.21. Αυτή η συνάρτηση δημιουργεί έναν τυχαίο αριθμό και αυτό δημιουργεί ένα crypto collectible το οποίο ανάλογα τον αριθμό εμφανίζει την αντίστοιχη εικόνα. Για τις ανάγκες της διπλωματικής για crypto collectibles χρησιμοποιήθηκαν εικόνες από κουτάβια και ο αριθμός τους περιορίστηκε σε τρία ενώ θα μπορούσαν να είναι θεωρητικά άπειρα.



Εικόνα 4.20 Crypto collectible που αποκτήθηκε μετά από νίκη

```
66          function createitem() private  {
67          int ran = randNum(3);
68              uint id1=items.push(item(ran,0,0 ether))-1;
69           itemtoowner[id1] = msg.sender;
70           itemCount[msg.sender]++;
71          }
```

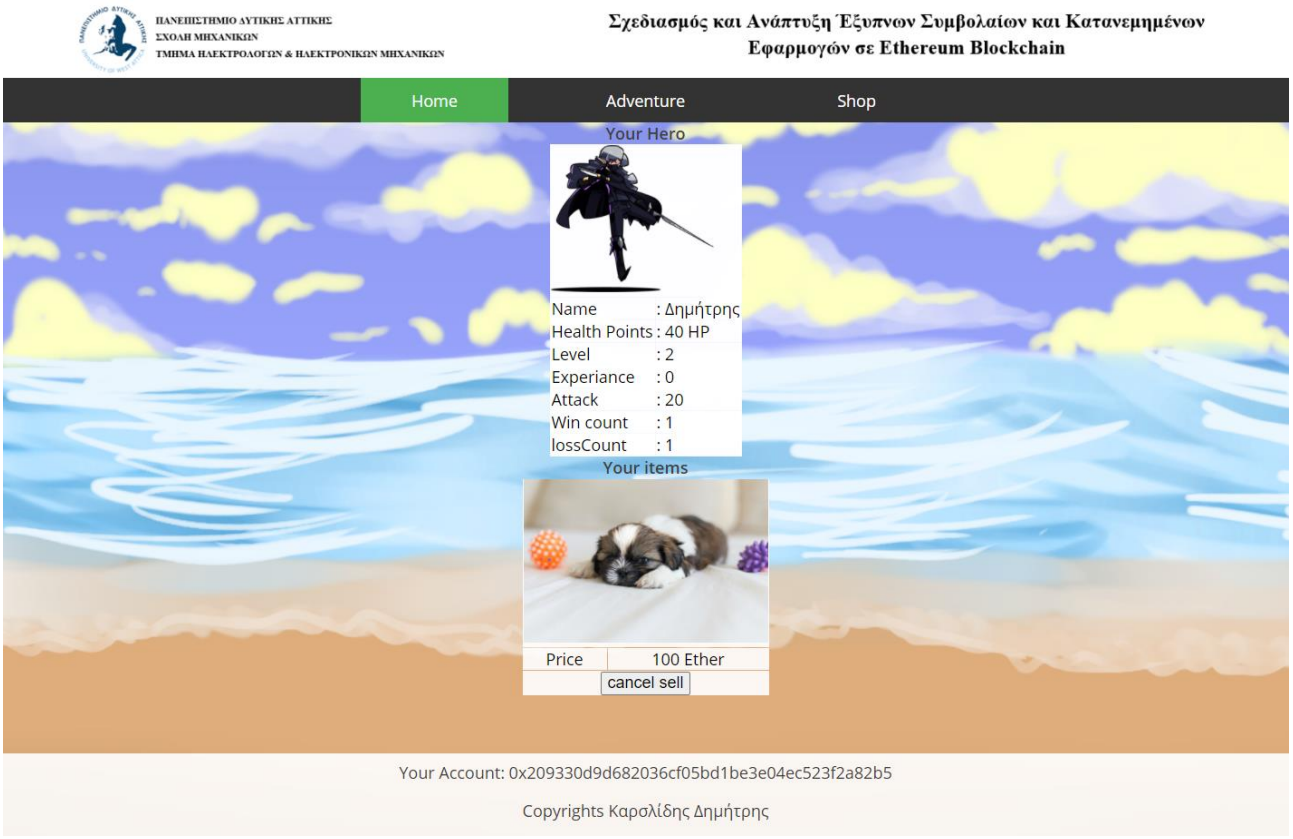
Εικόνα 4.21 Συνάρτηση δημιουργίας crypto collectible

Συνεχίζοντας με τις συναρτήσεις του έξυπνου συμβολαίου, έχουμε την δυνατότητα να αγοράζουμε και να πουλάμε crypto collectibles. Στην εικόνα 4.22 μπορούμε να δούμε μία φόρμα στην οποία μπορούμε να θέσουμε την τιμή που θέλουμε να πουλήσουμε το crypto collectible μας και μόλις κάνουμε submit καλείται η συνάρτηση της εικόνας, η οποία θέτει το crypto collectible μας προς πώληση.

```
84          function tosell(uint _itemid,uint _price){
85              item storage myitem = items[_itemid];
86            require(msg.sender == itemtoowner[_itemid]);
87            require(myitem.sellable ==0);
88              myitem.price= _price;
89              myitem.sellable = 1;
90          }
```

Εικόνα 4.22 Συνάρτηση για πώληση crypto collectible

Μόλις βάλουμε το crypto collectible μας προς πώληση έχουμε την δυνατότητα να ακυρώσουμε την πώληση του όπως φαίνεται την εικόνα 4.23 μέσω της συνάρτησης της εικόνας 4.24.

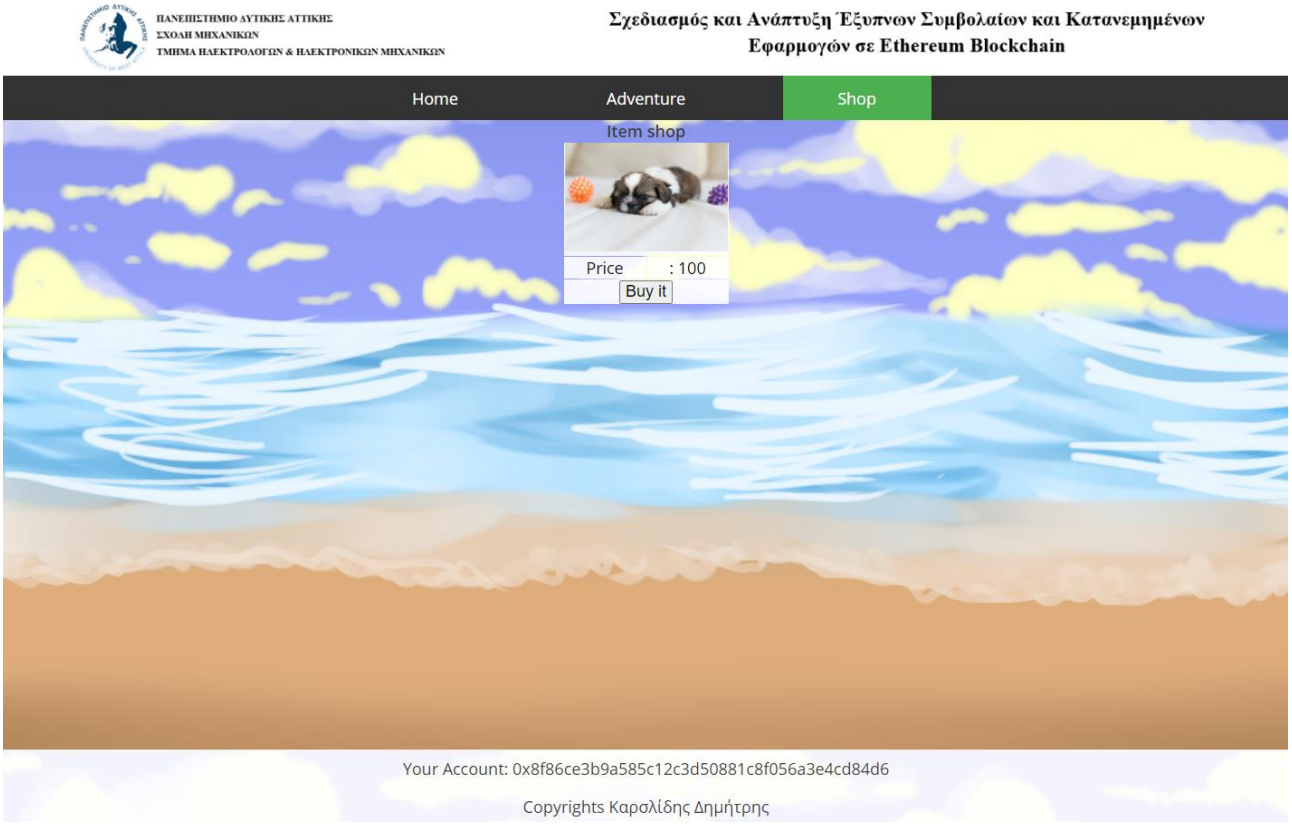


Εικόνα 4.23 User interface με δυνατότητα ακύρωση πώλησης

```
91          function cancelsell(uint _itemid){
92          require(msg.sender == itemtoowner[_itemid]);
93            item storage myitem = items[_itemid];
94             require(myitem.sellable ==1);
95            myitem.sellable = 0;
96        }
```

Εικόνα 4.24 Συνάρτηση για ακύρωση πώλησης

Επόμενό βήμα, να δούμε την αγορά αυτού του αντικειμένου από κάποιον άλλον χρήστη. Αρχικά μέσω του metamask συνδεόμαστε σε άλλον λογαριασμό και πηγαίνουμε στην σελίδα Shop (εικόνα 4.25). Εκεί έχουμε την δυνατότητα να αγοράσουμε τα crypto collectible που είναι προς πώληση μέσω της συνάρτησης transferFrom της εικόνας 4.26. Αυτή η συνάρτηση χρησιμοποιεί την αντίστοιχη συνάρτηση από το συμβόλαιο με το πρότυπο ERC721 και με την σειρά της καλεί την συνάρτηση buyit η οποία αλλάζει την διεύθυνση που αντιστοιχεί στο crypto collectible με αποτέλεσμα να αλλάζει η ιδιοκτησία με την προϋπόθεση ότι ο αγοραστής έχει πληρώσει το ανάλογο αντίτιμο και επίσης προσθέτει το αντίτιμο στο πορτοφόλι του πωλητή.

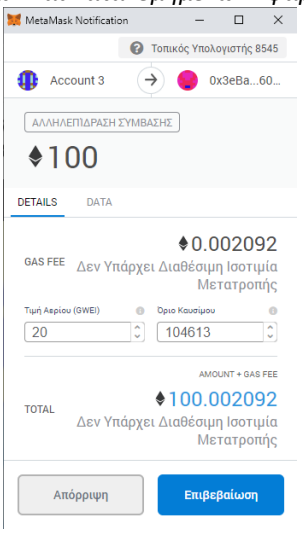


Εικόνα 4.25 E-shop για cryptocollectible

```
97          function buyit(address _from, address _to,uint _tokenId) private{
98              itemCount[_to]++;
99              itemCount[_from]--;
100             itemtoowner[_tokenId] = _to;
101             cancelsell(_tokenId);
102         }
103         function transferFrom(address _from, address _to, uint256 _tokenId) external payable{
104             item storage myitem = items[_tokenId];
105             uint priceof = myitem.price;
106         require(msg.value > priceof );
107         require(_from != _to);
108         require(myitem.sellable ==1);
109         _from.transfer(msg.value);
110          Transfer(_from, _to, _tokenId);
111         buyit( _from,  _to, _tokenId);
112     }
```

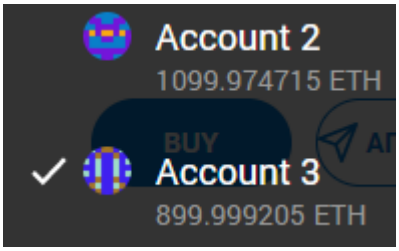
Εικόνα 4.26 συναρτήσεις για αγορά crypto collectible

Μόλις πατήσουμε το κουμπί για την αγορά του αντικειμένου το metamask θα επίσης παραπέμψει να πληρώσουμε την συναλλαγή επίσης εικόνας 4.27 όπου μπορούμε να προσέξουμε πως εκτός από τα τέλη επίσης συναλλαγής πρέπει να πληρώσουμε και την αξία του αντικειμένου.

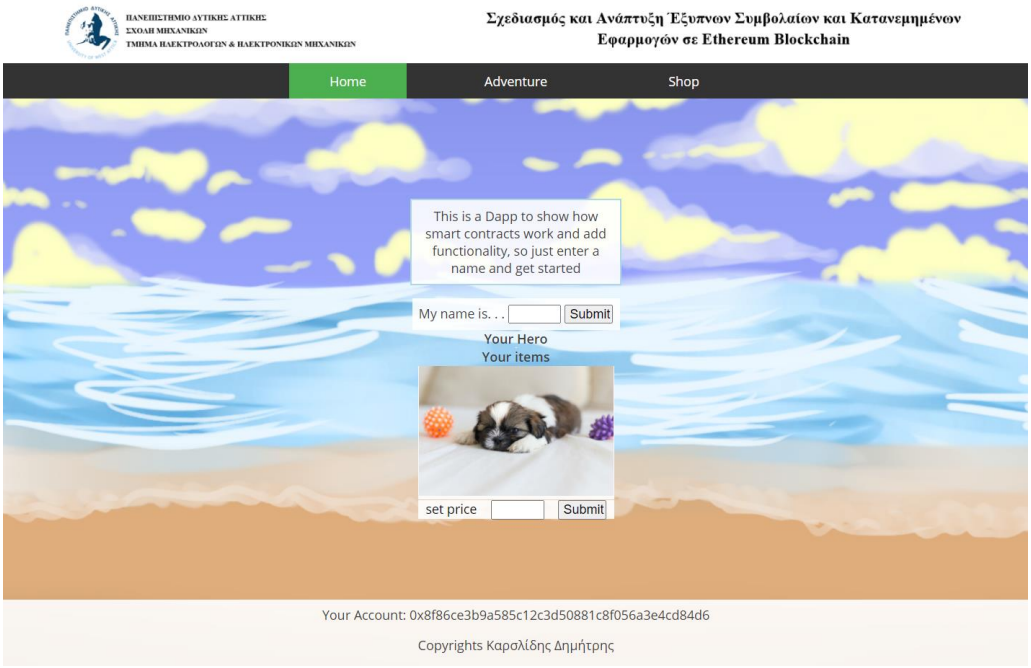


Εικόνα 4.27 Συναλλαγή για την αγορά crypto collectible

Μόλις ολοκληρώσουμε την συναλλαγή μπορούμε να δούμε καταρχήν στην εικόνα 4.28 ότι μεταφέρθηκαν από τον λογαριασμό 3 στον λογαριασμό 2 τα ether και μεταβιβάστηκε το crypto collectible από τον 2 στον 3 (εικόνα 4.29)



Εικόνα 4.28 Μεταφορά ETH μετά από συναλλαγή



Εικόνα 4.29 Μεταβιβασμένο crypto collectible

Επίσης, αν το επιτρέπει το smart contract, μπορεί κάποιο άλλο smart contract να διαβάσει τα δεδομένα του κύριου. Στο smart contract που υλοποιήθηκε προσθέσαμε την δυνατότητα να μπορούν να διαβαστούν τα δεδομένα των ηρώων μέσω επίσης συνάρτησης επίσης εικόνας 4.30.

```
220             function gethero(uint256 _id) external view returns (
221             string name,
222             int life,
223             int lvl,
224             int xp,
225             int attack,
226             int winCount,
227             int lossCount,
228             int herodmg
229             ){
230               hero storage kit = heroes[_id];
231               name = string(kit.name);
232               life = int(kit.life);
233               lvl = int(kit.lvl);
234               xp = int(kit.xp);
235               attack = int(kit.attack);
236               winCount = int(kit.winCount);
237               lossCount = int(kit.lossCount);
238               herodmg = int(kit.herodmg);
239
240             }
```

Εικόνα 4.30 Συνάρτηση για δυνατότητα διαβάσματος δεδομένων από άλλο έξυπνο συμβόλαιο

Για να το δείξουμε αυτό υλοποιήθηκε ένα δεύτερο Dapp με παρόμοιο τρόπο επίσης και το πρώτο. Αρχικά για συνδέσουμε το δεύτερο Dapp με το πρώτο προσθέτουμε ένα contract με τον κώδικα της εικόνας 4.31, επίσης για να μπορούμε να καλούμε την συνάρτηση με τα στοιχεία του ήρωα και ένα δεύτερο contract με την διεύθυνση του κύριου Dapp καθώς και επίσης λειτουργίες του δευτερεύοντος.

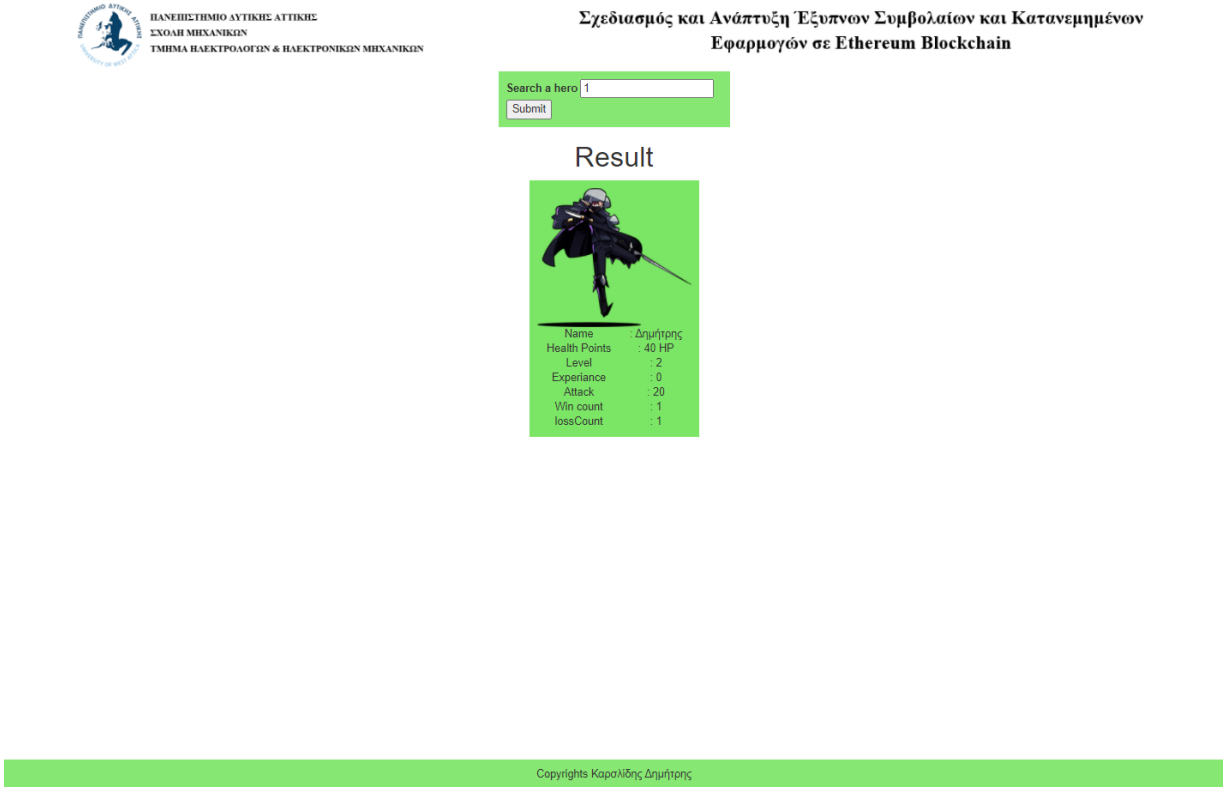
```
3     contract herointerface{
4         function gethero(uint256 _id) external view returns (
5             string name,
6             int life,
7             int lvl,
8             int xp,
9             int attack,
10            int winCount,
11            int trophies,
12            int lossCount,
13            int herodmg
14        );
15    }
```

Εικόνα 4.31 Συνάρτηση για διάβασμα δεδομένων από το πρώτο DApp

Στην εικόνα 4.32 μπορούμε να δούμε το UI του δεύτερου Dapp.

Εικόνα 4.32 Συνάρτηση για διάβασμα δεδομένων από το πρώτο DApp

Στην φόρμα μπορούμε να βάλουμε το id κάποιου ήρωα και να επίσης τον εμφανίσει. Θα βάλουμε την τιμή 1 και θα έχουμε το αποτέλεσμα επίσης εικόνα 4.33. Αυτή η συνάρτηση δεν κόστισε καθόλου gas διότι το διάβασμα των δεδομένων δεν «γράφει» κάτι στο blockchain αλλά ούτε χρειάζεται αποθηκευτικό χώρο.



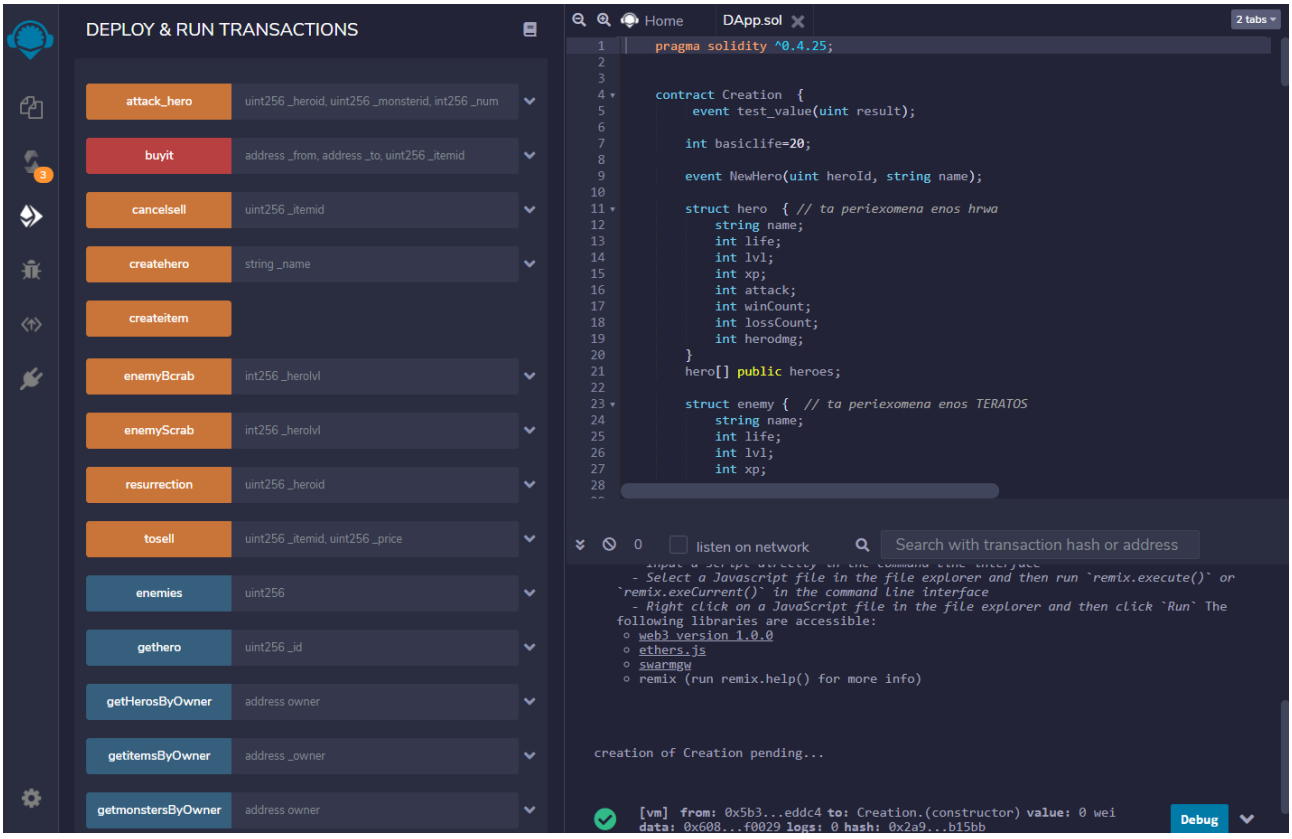
Εικόνα 4.33 Διάβασμα δεδομένων του πρώτου DApp από το δεύτερο

Ακόμη στην εικόνα 4.34 βλέπουμε μερικές από τις συναρτήσεις που ορίζει το πρότυπο ERC-721 token. Σαν μελλοντική βελτίωση, θα μπορούσαν να φτιαχτούν και οι υπόλοιπες συναρτήσεις του ERC-721.

```
239             event Approval(address indexed _owner, address indexed _approved, uint256 _tokenId);
240
241         function balanceOf(address _owner)  returns (uint256) {
242         return itemCount[_owner];
243         }
244         function ownerOf(uint256 _tokenId)  returns (address) {
245             address owner = itemtoowner[_tokenId];
246          return owner;
247         }
248         function approve(address _to, uint256 _tokenId){
249         address owner = ownerOf(_tokenId);
250         require(msg.sender == owner );
251          require(msg.sender != _to);
252         allowed[msg.sender][_to] = _tokenId;
253         Approval(msg.sender, _to, _tokenId);
254         }
255     }
```

Εικόνα 4.34 Συναρτήσεις προτύπου ERC-721

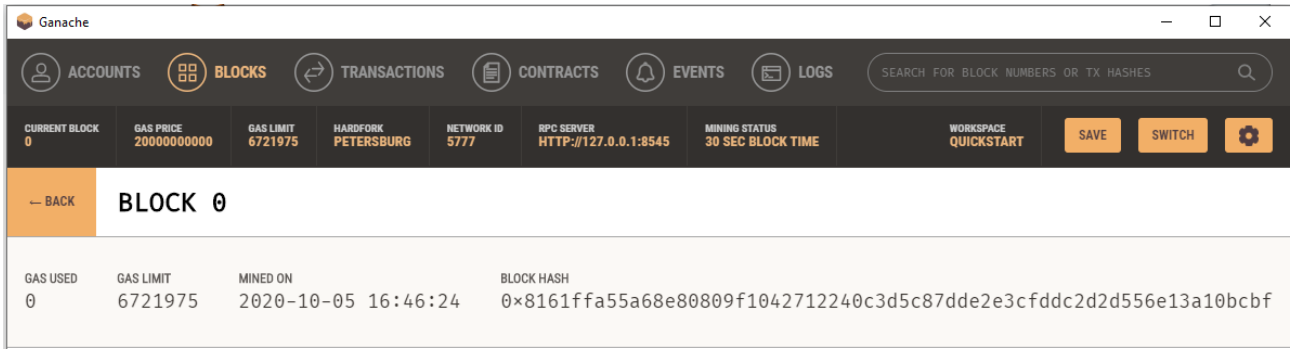
Λόγο ότι, ότι γράφετε στο blockchain παραμένει εκεί για πάντα είναι πολύ σημαντικό να ελέγξουμε ότι τα smart contract επίσης λειτουργούν σωστά. Για τον έλεγχο επίσης λειτουργικότητας των smart contracts χρησιμοποιήσαμε την ιστοσελίδα https://remix.ethereum.org/ εκεί δημιουργήσαμε ένα .sol αρχείο με τον κώδικα επίσης, κάναμε compile και deploy. Έτσι έχουμε το UI επίσης εικόνας 4.35 και χρησιμοποιώντας επίσης συναρτήσεις μέσω του remix μπορούμε να ελέγξουμε αν οι συναρτήσεις επίσης λειτουργούν επίσης θα έπρεπε.



Εικόνα 4.35 User interface του Remix

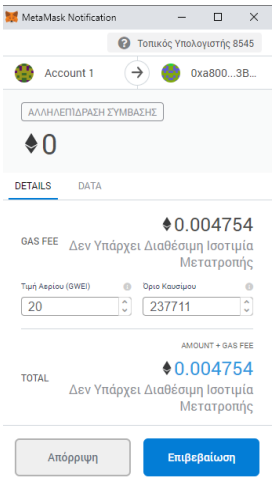
Επίσης παρατηρήσεις που μπορούμε να κάνουμε είναι οι εξής.

Η δημιουργία του μπλοκ γένεσης στη εικόνα 4.36.



Εικόνα 4.36 Μπλοκ γένεσης στο ganache

Η πληρωμή gas κάθε φορά που καλείται μία συνάρτηση λόγο ότι γράφει στο Ethereum blockchain εικόνα στην εικόνα 4.37.



Εικόνα 4.37 πληρωμή Gas μετά από κλήση συνάρτησης για δημιουργία εχθρού

Η δομή επίσης μπλοκ του Ethereum, την οποία μπορούμε να δούμε μέσο των logs του ganache στην εικόνα 4.38. Να σημειωθεί πώς κάποιες πληροφορίες όπως το ύψος του μπλοκ (number 0x12) και το hash δεν περιέχονται στην δομή του μπλοκ, το ganache όμως τα δείχνει.

```
  "number": "0x12",
  "hash": "0x10585df5ff9dec2a32d106fc7c1cff1f3a03935554d927d115e91017a13c636d",
  "parentHash": "0x70c14ba0e41908fa65b31eb7e7d27743f174bdd030d8a0a40150e1675f83733f",
  "mixHash": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "nonce": "0x0000000000000000",
  "sha3Uncles": "0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347",
  "logsBloom": "0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
               00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
               00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
               00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",
  "transactionsRoot": "0x80e5ff7649aaf9ebed4f65f1684e14ae0b0d46b506faef833606429e4281c15f",
  "stateRoot": "0x4460f7af7b0d410ff46abe687d99151e0faef853db4d6e805dad03de34279529",
  "receiptsRoot": "0xc4881a1dd0bb674e161fd859e68d752bacead4d21629452dbdfedbe00b58db97",
  "miner": "0x0000000000000000000000000000000000000000",
  "difficulty": "0x0",
  "totalDifficulty": "0x0",
  "extraData": "0x",
  "size": "0x3e8",
  "gasLimit": "0x6691b7",
  "gasUsed": "0x2e954",
  "timestamp": "0x5f7b33ee",
  "transactions": [
    {
      "hash": "0xd23e4c1934487bf18dfd90e8a70af3260e4313b0217ca8e89e383b208a168d6e",
      "nonce": "0x1",
      "blockHash": "0x10585df5ff9dec2a32d106fc7c1cff1f3a03935554d927d115e91017a13c636d",
      "blockNumber": "0x12",
      "transactionIndex": "0x0",
      "from": "0x10a87369e70d6c8f58dd3afd8291fc4698b68ed0",
      "to": "0x1b9abfcb927d2a66095fc1c900376e18b0b477cb",
      "value": "0x0",
      "gas": "0x45dfe",
      "gasPrice": "0x4a817c800",
      "input": "0x72039867000000000000000000000000000000000000000000000000000000000000002000000000000000000000000000000000000000000000000000000000000000010ce94ceb7cebcceaecf
               84cf81ceb7cf8200000000000000000000000000000000",
      "v": "0x2d46",
      "r": "0x94ee6fe47f722bc128519b880cc9a06fa32af98be05c43499167f7d6f3c72507",
      "s": "0x3a4dc3780413f0d58913d92df1b5dab1a2caf92e8ea4ca21ca46d5a0fd07ec30"
    }
  ],
  "uncles": []
}
```


Εικόνα 4.38 Δομή επίσης Ethereum μπλοκ

Επίσης θα αναφερθεί σαν παράδειγμα ένα σφάλμα που θα μπορούσε να δημιουργήσει πρόβλημα στο συμβόλαιο μας. Η συνάρτηση της εικόνας μας δημιουργεί ένα αντικείμενο και έχει οριστεί σαν private έτσι ώστε μόνο το συμβόλαιο να μπορεί να την καλέσει, αν αυτή η συνάρτηση είχε οριστεί σαν public λόγο απροσεξίας, κάθε χρήστης θα μπορούσε να την καλέσει αποκτώντας σχεδόν τσάμπα (πληρώνοντας μόνο το gas) τα crypto collectibles καθιστώντας τα ενδεχόμενος άχρηστα λόγο του υπερπληθυσμού και τελικά να αχρήστευση το συμβόλαιο μας.

Ακόμα μέσω των εικόνων 4.39 και 4.40 μπορούμε να δούμε το gas που χρειάστηκε για το κύριος dapp και το δευτερεύον αντίστοιχα, μπορούμε να παρατηρήσουμε ότι το πρώτο dapp κοστίζει πολύ περισσότερο από το δεύτερο λόγω ότι είναι πιο περίπλοκο και απαιτεί περισσότερο αποθηκευτικό χώρο στο blockchain.

```
Summary
=======
> Total deployments:   1
> Final cost:          0.05171652 ETH
```

Εικόνα 4.39 Κόστος για το πρώτο DApp

```
Summary
=======
> Total deployments:   1
> Final cost:          0.00900124 ETH
```

Εικόνα 4.40 Κόστος για το δεύτερο DApp

5 ΣΥΜΠΕΡΑΣΜΑΤΑ

Συμπερασματικά όσον αφορά το θεωρητικό κομμάτι, μάθαμε πώς το blockchain δίνει τον αποκλειστικό έλεγχο των συναλλαγών στους χρήστες, μέσω των δημόσιων-ιδιωτικών κλειδιών και αυτοί αντιπροσωπεύονται μέσω των διευθύνσεων στο blockchain. Επίσης είδαμε τι πληροφορίες περιλαμβάνουν τα μπλοκ του blockchain του Ethereum και πως αυτές συνοψίζονται με την χρήση των δέντρων merkle. Μάθαμε επίσης πως δημιουργούνται αυτά τα μπλοκ από τους miners και για την διαδικασία του mining, με την οποία τα μπλοκ επαληθεύονται με την χρήση του αλγόριθμου συναίνεσης PoW και προστίθενται στην αλυσίδα.

Ακόμα Επιχειρηματολογήσαμε για τους λόγους που υπερισχύει το blockchain του Ethereum έναντι άλλων υλοποιήσεων blockchain(π.χ Bitcoin), με την χρήση του παγκόσμιου αποκεντρικοποιημένου υπολογιστή του EVM στον οποίο εκτελούνται προγράμματα που έχουν αναπτυχθεί στο περιβάλλον του Ethereum και πως με την χρήση του gas αποτρέπεται η κακόβουλη χρήση του. Επιπρόσθετα μάθαμε τι είναι τα έξυπνα συμβόλαια και πως γράφονται, τα οποία είναι βασικό χαρακτηριστικό του Ethereum και επιτρέπει στους προγραμματιστές την ανάπτυξη συμβολαίων και εφαρμογών, μάθαμε πως τα συμβόλαια αυτά ανεβαίνουν στο δίκτυο του blockchain και ύστερα πως καλούνται, είτε αυτόματα όταν εκπληρωθούν οι προϋποθέσεις του συμβολαίου είτε από κάποιον χρήστη μέσω κάποιας συναλλαγής, καθώς και πώς μπορούμε να τα διαγράψουμε όταν αυτά πλέον δεν χρειάζονται, όπως για παράδειγμα τις εφαρμογές ψηφοφορίας. Επίσης μάθαμε ότι λόγω της αμεταβλητότητας των έξυπνων συμβολαίων πρέπει να δοθεί ιδιαίτερη προσοχή στον κώδικα και να ακολουθηθούν κάποιες καλές πρακτικές για την αποφυγή κενών ασφαλείας. Παρόλο που τα έξυπνα συμβόλαια είναι αμετάβλητα, είδαμε τρόπους με τους οποίους μπορούμε να κάνουμε αναβαθμίσεις σε αυτά, όπως με την χρήση συμβολαίων αφέντη-σκλάβου. Μάθαμε επίσης τι είναι τα DApps και ποια στοιχεία του μπορούν να αποκεντρικοποιηθούν και πώς χρησιμοποιώντας το blockchain σαν backend σε συνδυασμό με τεχνολογίες για frontend, κάνουν πρόσβαση και την χρήση αυτών των εφαρμογών εύκολή για όλους τους χρήστες.

Από το κομμάτι της υλοποίησης, είδαμε μέσω των εικόνων 4.10, 4.12, 4.13, 4.14, 4.16, 4.17, 4.18, 4.20, 4.23, 4.25 και 4.33 πως οι χρήστες μπορούν να καλέσουν τις συναρτήσεις του συμβολαίου που υπάρχει στο blockchain του Ethereum καθώς και την αυτόματη κλήση συνάρτησης όταν πληρούνται οι προϋποθέσεις όπως στην περίπτωση της εικόνας 4.19 και τέλος είδαμε πως είναι δυνατές οι συναλλαγές για αντικείμενα αξίας, όπως τα crypto collectibles μέσο ενός έξυπνου συμβολαίου όπως μπορούμε να δούμε στις εικόνες 4.20, 4.23 και 4.25.

Η δημιουργία έξυπνων συμβολαίων και κατ' επέκταση αποκεντρικοποιημένων εφαρμογών (Dapps) είναι από εύκολη έως και σύνθετη ανάλογα την εφαρμογή που θέλει να φτιάξει ο καθένας. Επιπλέον η εκμάθηση της γλώσσας προγραμματισμού συμβολαίων solidity ήταν ομαλή, καθώς μοιάζει σε μεγάλο βαθμό στην JavaScript και είναι πολύ ενδιαφέρουσα λόγω του μεγάλου εύρους εφαρμογών και συμβολαίων που μπορεί να υλοποιήσει κάνεις, έχοντας σαν όριο μόνο την φαντασία (και το gas!).

Βιβλιογραφία – Αναφορές - Διαδικτυακές Πηγές

1. *Alharby, M., Moorsel, A. (2017). Blockchain Based Smart Contracts : A Systematic Mapping Study. Computer Science & Information Technology (CS & IT). doi: 10.5121/csit.2017.71011*
2. *Antonopoulos, A. (2018). Mastering Bitcoin. Sebastopol, CA: O'Reilly.*
3. *Antonopoulos, A., Wood, G. (2019). Mastering Ethereum. O'Reilly Media, Inc.*
4. *Anwar, H. (2019). The Ultimate Guide to Pros and Cons of Blockchain | 101 Blockchains. Retrieved 26 January 2021, from https://101blockchains.com/pros-and-cons-of-blockchain/*
5. *Bashir, I. (2018). MASTERING BLOCKCHAIN (2nd ed.). [Place of publication not identified]: PACKT Publishing.*
6. *Bashir, I. (2020). Mastering Blockchain (3rd ed.). [S.l.]: Packt Publishing.Jani, Shailak. (2018). An Overview of Ethereum & Its Comparison with Bitcoin.*
7. *Bauerle, N. (2020). Blockchain 101 - CoinDesk. Retrieved 26 January 2021, from https://www.coindesk.com/learn/blockchain-101/what-is-a-distributed-ledger*
8. *Blockchain Advantages and Disadvantages | Binance Academy. (2020). Retrieved 26 January 2021, from https://academy.binance.com/en/articles/positives-and-negatives-of-blockchain*
9. *Bouchefra, A. (2018). Remix: Develop Smart Contracts for the Ethereum Blockchain - SitePoint. Retrieved 18 November 2020, from https://www.sitepoint.com/remix-smart-contracts-ethereum-blockchain/*
10. *Comparison - Centralized, Decentralized and Distributed Systems - GeeksforGeeks. (2019). Retrieved 26 January 2021, from https://www.geeksforgeeks.org/comparison-centralized-decentralized-and-distributed-systems/*
11. *di Angelo, M., & Salzer, G. (2020). Tokens, Types, and Standards: Identification and Utilization in Ethereum. 2020 IEEE International Conference On Decentralized Applications And Infrastructures (DAPPS). doi: 10.1109/dapps49028.2020.00001*
12. *Gates, M. (2017). Blockchain. North Charleston: CreateSpace.*
13. *Getting started with the Web - Learn web development | MDN. (2020). Retrieved 26 January 2021, from https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web*
14. *Golosova, J., & Romanovs, A. (2018). The Advantages and Disadvantages of the Blockchain Technology. 2018 IEEE 6Th Workshop On Advances In Information, Electronic And Electrical Engineering (AIEEE). doi: 10.1109/aieee.2018.8592253*
15. *Goyal, S. (2015). Centralized vs Decentralized vs Distributed. Retrieved 26 January 2021, from https://medium.com/delta-exchange/centralized-vs-decentralized-vs-distributed-41d92d463868*
16. *Gupta, M. (2018). How to make smart contracts upgradable! | Hacker Noon. Retrieved 3 December 2020, from https://hackernoon.com/how-to-make-smart-contracts-upgradable-2612e771d5a2*
17. *History of Blockchain | Binance Academy. (2021). Retrieved 26 January 2021, from https://academy.binance.com/en/articles/history-of-blockchain*
18. *Huang, Y., Bian, Y., Li, R., Zhao, J., & Shi, P. (2019). Smart Contract Security: A Software Lifecycle Perspective. IEEE Access, 7, 150184-150202. doi: 10.1109/access.2019.2946988*

19. Mahato, S., Khatua, T., Das, A., & Chowdhury, T. (2019). A Brief Discussion on Two Different Cryptocurrency: BITCOIN & ETHEREUM. International Journal Of Computer Trends And Technology, 67(9), 32-38. doi: 10.14445/22312803/ijctt-v67i9p106
20. MetaMask. (2021). Retrieved 26 January 2021, from https://metamask.io/
21. Mohanta, B., Panda, S., & Jena, D. (2018). An Overview of Smart Contract and Use Cases in Blockchain Technology. 2018 9Th International Conference On Computing, Communication And Networking Technologies (ICCCNT). doi: 10.1109/icccnt.2018.8494045
22. Narayanan, A., Bonneau, J., Felten, E., Miller, A., & Goldfeder, S. (2016). Bitcoin and Cryptocurrency Technologies. doi: 10.1016/s1353-4858(16)30074-5
23. Niranjanamurthy, M., Nithya, B., & Jagannatha, S. (2018). Analysis of Blockchain technology: pros, cons and SWOT. Cluster Computing, 22(S6), 14743-14757. doi: 10.1007/s10586-018-2387-5
24. npm | npm Docs. Retrieved 24 November 2020, from https://docs.npmjs.com/cli/v6/commands/npm
25. Nzuva, S. (2019). Smart Contracts Implementation, Applications, Benefits, and Limitations. Journal Of Information Engineering And Applications. doi: 10.7176/jiea/9-5-07
26. Palladino, S. (2020). ETHEREUM FOR WEB DEVELOPERS. APRESS.
27. Prusty, N. (2017). Building Blockchain projects. Packt Publishing Ltd.
28. Rathod, N., Motwani, D. (2018). Security threats on Blockchain and its countermeasures. International Research Journal Of Engineering And Technology (IRJET), 5(11)
29. Rijmenam, M. (2017). How Blockchain Will Bring Back Data Ownership to Consumers. Retrieved 26 January 2021, from https://www.linkedin.com/pulse/how-blockchain-bring-back-data-ownership-consumers-mark-van-rijmenam
30. Sahu, M. (2020). What is Truffle Suite? Features, How to Install, How to Run Smart Contracts | upGrad blog. Retrieved 26 January 2021, from https://www.upgrad.com/blog/what-is-truffle-suite/
31. Sheikh, H. (2019). Smart Contract Development, Adoption and Challenges: The Powered Blockchain. International Research Journal Of Advanced Engineering And Science.
32. Sherman, A., Javani, F., Zhang, H., & Golaszewski, E. (2019). On the Origins and Variations of Blockchain Technologies. IEEE Security & Privacy, 17(1). doi: 10.1109/msec.2019.2893730
33. Srivastav, K. (2019). A Guide to Blockchain Immutability and Challenges - DZone Security. Retrieved 26 October 2020, from https://dzone.com/articles/a-guide-to-blockchain-immutability-and-chief-chall
34. What Is Solidity? What Are Its Use Cases? - Top Digital Agency. (2019). Retrieved 26 January 2021, from https://topdigital.agency/what-is-solidity-what-are-its-use-cases/

Παράρτημα Α : Κώδικας κύριου έξυπνου συμβολαίου

```
    pragma solidity ^0.4.25;

   import "./ERC721.sol";
    contract Creation  {
     event Transfer(address indexed _from, address indexed _to, uint256 indexed _to
kenId);

        int basiclife=20;

        event NewHero(uint heroId, string name);

        struct hero  { // ta periexomena enos hrwa
            string name;
            int life;
            int lvl;
            int xp;
            int attack;
            int winCount;
            int lossCount;
            int herodmg;
        }
        hero[] public heroes;

        struct enemy {  // ta periexomena enos TERATOS
            string name;
            int life;
            int lvl;
            int xp;
            int attack;
            int mondmg;
        }
        enemy[] public enemies;

        struct item  {
            int uid;
            int sellable;
            uint price;
        }
        item[] public items;

      mapping (uint => address) public itemtoowner;
      mapping (uint => address) public herotoowner;
      mapping (address => uint) public  heroCount;
      mapping (address => uint) public  itemCount;
      mapping (uint => address) public monstertoowner;
      mapping(address => mapping (address => uint256)) public allowed;
      uint public monsterCount;
```

```
        function createhero(string memory _name) public  {              //function
to create a hero
         require(heroCount[msg.sender] == 0);                            //user can
create a hero only if he hasen't one
           uint id= heroes.push(hero(_name,basiclife,1,0,10,0,0,0))-
1;  //add hero in the array of heroes
           herotoowner[id] = msg.sender;                                 //maps the
id with the address of the user that created him
           heroCount[msg.sender]++;                                      //
           emit NewHero(id, _name);
        }
         function enemyScrab(int _herolvl) public  {
           uint id=enemies.push(enemy("Small crab",15*_herolvl,_herolvl,7*_herolvl,
5*(_herolvl*_herolvl),0))-1;
           monstertoowner[id] = msg.sender;
           monsterCount++;
        }
        function enemyBcrab(int _herolvl) public  {
           uint id=enemies.push(enemy("Big crab",20*_herolvl,_herolvl,14*_herolvl,5
*(_herolvl*_herolvl),0))-1;
           monstertoowner[id] = msg.sender;
           monsterCount++;
        }

          function createitem() public  {
          int ran = randNum(3);
              uint id1=items.push(item(ran,0,0 ether))-1;
           itemtoowner[id1] = msg.sender;
           itemCount[msg.sender]++;
          }

           function getitemsByOwner(address _owner) external view returns( uint[] m
emory ) { //sinartisi gia euresi id kai emfanisi tou hrwa
             uint[] memory result = new uint[](itemCount[_owner]);
            uint counter = 0;
            for (uint i = 0; i < items.length; i++) {
            if (itemtoowner[i] == _owner) {
              result[counter] = i;
              counter++;
                   }
                  }
              return result;
            }
            function tosell(uint _itemid,uint _price) public{
                item storage myitem = items[_itemid];
              require(msg.sender == itemtoowner[_itemid] || _itemid == allowed[ms
g.sender][msg.sender]);
              require(myitem.sellable ==0);
                myitem.price= _price;
                myitem.sellable = 1;
            }
              function cancelsell(uint _itemid) public{
```

```
                require(msg.sender == itemtoowner[_itemid]);
                  item storage myitem = items[_itemid];
                   require(myitem.sellable ==1);
                  myitem.sellable = 0;
              }
               function buyit(address _from, address _to,uint _tokenId) private{
                  itemCount[_to]++;
                  itemCount[_from]--;
                  itemtoowner[_tokenId] = _to;
                  cancelsell(_tokenId);
              }
               function transferFrom(address _from, address _to, uint256 _tokenId)
external payable{
                    item storage myitem = items[_tokenId];
                    uint priceof = myitem.price;
                  require(msg.value > priceof );
                  require(_from != _to);
                  require(myitem.sellable ==1);
                  _from.transfer(msg.value);
                   Transfer(_from, _to, _tokenId);
                  buyit( _from,  _to, _tokenId);
                }
            function retarrray() external view returns (uint[] memory) {
                if (items.length <7 ){
                     uint[] memory result = new uint[](items.length);
                     for (uint i = 0; i < 7; i++) {
                  if (i ==items.length){
                    return result;
                  }
                  result[i] = i;
                }
                 return result;
                } else {
                     uint[] memory result2 = new uint[](7);
                     for (uint j = 0; j < 7; j++) {
                  if (j ==items.length){
                    return result2;
                  }
                  result2[j] = j;
                }
                 return result2;
                }
            }
           function getHerosByOwner(address owner) external view returns( uint ) {
//sinartisi gia euresi id kai emfanisi tou hrwa
            uint result =999999;
            for (uint i = 0; i < heroes.length; i++) {
                  if (herotoowner[i] == owner) {
                    result = i;
                    break;
                    }
                    }
```

```
            return result;
            }
            function getmonstersByOwner(address owner) external view returns( uint
) { //sinartisi gia euresi id kai emfanisi tou teratos
            uint result =0;
            for (uint i = enemies.length ; i > 0 ; i--) {
                  if (monstertoowner[i] == owner) {
                    result = i;
                    break;
              }
            }
            return result;
            }

            uint nonce=0;

       function randNum(int _num) private   returns(int) {  //sinartisi gia tixaies
 times
        int randomnumber = int(keccak256(abi.encodePacked(now, msg.sender, nonce)))
 % _num;
        if (randomnumber <=0) {
            randomnumber=(-1)*randomnumber;
        }
        nonce++;
        return randomnumber;
        }
          function randNum2(uint _num) external  view returns(uint) {  //sinartisi
gia tixaies times
        uint randomnumber = uint(keccak256(abi.encodePacked(now, msg.sender, nonce)
)) % _num;

        nonce++;
        uint result = randomnumber;
        return result;
        }


        function attack_hero(uint _heroid, uint _monsterid, int _num  ) external  {


          hero storage myhero = heroes[_heroid];
          enemy storage monster = enemies[_monsterid];
          int attackofhero=0;

          int attackofmonster=randNum(monster.attack) + 5;
          if(_num==0 ) {
           attackofhero=randNum(myhero.attack) +5;
          }

            monster.life=monster.life - attackofhero;

          if(monster.life>0){
```

```
            myhero.life=myhero.life - attackofmonster;
          }else{
            myhero.life=myhero.life;
          }
          myhero.herodmg=attackofhero;
          monster.mondmg=attackofmonster;

          if (myhero.life <= 0 ) {
           myhero.lossCount++;
          }
          if (monster.life <= 0 && myhero.life >= 1) {
            myhero.winCount++;
            myhero.xp+=monster.xp;
            if (randNum(100) > 0 ){
                   createitem();
            }
            if (myhero.xp>=10+2*myhero.lvl){
              myhero.lvl++;
              myhero.xp=0;
              myhero.life=basiclife*myhero.lvl;
              myhero.attack=10*myhero.lvl;
            }
         }
        }
        function resurrection(uint _heroid) public {
             hero storage myhero = heroes[_heroid];
             myhero.life=basiclife*myhero.lvl;
        }


            function gethero(uint256 _id) external view returns (
           string name,
           int life,
           int lvl,
           int xp,
           int attack,
           int winCount,
           int lossCount,
           int herodmg
           ){
             hero storage kit = heroes[_id];
             name = string(kit.name);
             life = int(kit.life);
             lvl = int(kit.lvl);
             xp = int(kit.xp);
             attack = int(kit.attack);
             winCount = int(kit.winCount);
             lossCount = int(kit.lossCount);
             herodmg = int(kit.herodmg);

           }
```

```
                event Approval(address indexed _owner, address indexed _approved,
uint256 _tokenId);

                function balanceOf(address _owner)  returns (uint256) {
                return itemCount[_owner];
                }
                function ownerOf(uint256 _tokenId)  returns (address) {
                    address owner = itemtoowner[_tokenId];
                 return owner;
                }
                function approve(address _to, uint256 _tokenId){
                address owner = ownerOf(_tokenId);
                require(msg.sender == owner );
                 require(msg.sender != _to);
                allowed[msg.sender][_to] = _tokenId;
                Approval(msg.sender, _to, _tokenId);
                }
   }
```

Παράρτημα Β : Κώδικας δευτέρου έξυπνου συμβολαίου

```
    pragma solidity ^0.4.25;

    contract herointerface{
        function gethero(uint256 _id) external view returns (
            string name,
            int life,
            int lvl,
            int xp,
            int attack,
            int winCount,
            int lossCount,
            int herodmg
      );
    }
    contract Creation2{
      address ckAddress = 0x6170Fbd2b8346E57AB7269232d25DA20853D670d;
herointerface Creation = herointerface(ckAddress);

      function dosomething(uint _heroid) external view returns( string) {
        string memory name;


              (name,,,,,,,) = Creation.gethero(_heroid);
              return name;
      }
      function dosomething2(uint _heroid) external view returns(int[6]) {

            int life;
            int lvl;
            int xp;
            int attack;
            int winCount;
            int lossCount;

             (,life,lvl,xp,attack,winCount,lossCount,) = Creation.gethero(_heroid);
             return [life,lvl,xp,attack,winCount,lossCount];
      }
    }
```