ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Ανάπτυξη web3 εφαρμογής ηλεκτρονικού καταστήματος

Γεώργιος Παπαδόπουλος

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΥΠΕΥΘΥΝΟΣ

Γεώργιος Σπαθούλας
Μέλος ΕΔΙΠ

Λαμία έτος 2022

ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Ανάπτυξη web3 εφαρμογής ηλεκτρονικού καταστήματος

Γεώργιος Παπαδόπουλος

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΥΠΕΥΘΥΝΟΣ

Γεώργιος Σπαθούλας
Μέλος ΕΔΙΠ

Λαμία έτος 2022

UNIVERSITY OF THESSALY

SCHOOL OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE & TELECOMMUNICATIONS

Development web3 e-shop application

George Papadopoulos

FINAL THESIS

ADVISOR

George Spathoulas
LTS Member

Lamia year 2022

«Με ατομική μου ευθύνη και γνωρίζοντας τις κυρώσεις [(1)], που προβλέπονται από της διατάξεις της παρ. 6 του άρθρου 22 του Ν. 1599/1986, δηλώνω ότι:

*1. Δεν παραθέτω κομμάτια βιβλίων ή άρθρων ή εργασιών άλλων αυτολεξεί **χωρίς να τα περικλείω σε εισαγωγικά** και χωρίς να αναφέρω το συγγραφέα, τη χρονολογία, τη σελίδα. Η αυτολεξεί παράθεση χωρίς εισαγωγικά χωρίς αναφορά στην πηγή, είναι λογοκλοπή. Πέραν της αυτολεξεί παράθεσης, λογοκλοπή θεωρείται και η παράφραση εδαφίων από έργα άλλων, συμπεριλαμβανομένων και έργων συμφοιτητών μου, καθώς και η παράθεση στοιχείων που άλλοι συνέλεξαν ή επεξεργάσθηκαν, χωρίς αναφορά στην πηγή. Αναφέρω πάντοτε με πληρότητα την πηγή κάτω από τον πίνακα ή σχέδιο, όπως στα παραθέματα.*

*2. Δέχομαι ότι η αυτολεξεί **παράθεση χωρίς εισαγωγικά**, ακόμα κι αν συνοδεύεται από αναφορά στην πηγή σε κάποιο άλλο σημείο του κειμένου ή στο τέλος του, είναι αντιγραφή. Η αναφορά στην πηγή στο τέλος π.χ. μιας παραγράφου ή μιας σελίδας, δεν δικαιολογεί συρραφή εδαφίων έργου άλλου συγγραφέα, έστω και παραφρασμένων, και παρουσίασή τους ως δική μου εργασία.*

3. Δέχομαι ότι υπάρχει επίσης περιορισμός στο μέγεθος και στη συχνότητα των παραθεμάτων που μπορώ να εντάξω στην εργασία μου εντός εισαγωγικών. Κάθε μεγάλο παράθεμα (π.χ. σε πίνακα ή πλαίσιο, κλπ), προϋποθέτει ειδικές ρυθμίσεις, και όταν δημοσιεύεται προϋποθέτει την άδεια του συγγραφέα ή του εκδότη. Το ίδιο και οι πίνακες και τα σχέδια

4. Δέχομαι όλες τις συνέπειες σε περίπτωση λογοκλοπής ή αντιγραφής.

Ημερομηνία: ……/…../20……

Ο – Η Δηλ.

(1) «Όποιος εν γνώσει του δηλώνει ψευδή γεγονότα ή αρνείται ή αποκρύπτει τα αληθινά με έγγραφη υπεύθυνη δήλωση
του άρθρου 8 παρ. 4 Ν. 1599/1986 τιμωρείται με φυλάκιση τουλάχιστον τριών μηνών. Εάν ο υπαίτιος αυτών των πράξεων
σκόπευε να προσπορίσει στον εαυτόν του ή σε άλλον περιουσιακό όφελος βλάπτοντας τρίτον ή σκόπευε να βλάψει άλλον, τιμωρείται με κάθειρξη μέχρι 10 ετών.»

ΠΕΡΙΛΗΨΗ

Η τεχνολογία blockchain ουσιαστικά μπορεί να περιγραφεί ως ένα κατανεμημένο δίκτυο λογισμικού το οποίο λειτουργεί ως ένας ψηφιακός μηχανισμός που επιτρέπει την ασφαλή μεταφορά περιουσιακών στοιχείων χωρίς μεσάζοντα (π.χ. τράπεζα). Όπως το διαδίκτυο είναι μια τεχνολογία που διευκολύνει την ψηφιακή ροή πληροφοριών, έτσι και το blockchain είναι μια τεχνολογία που διευκολύνει την ψηφιακή συναλλαγή τεκμηρίων με κάποια αξία. Αυτά τα τεκμήρια θα μπορούσαν να είναι οτιδήποτε, από πραγματικά νομίσματα, έως γη. Επιπλέον μπορούν να αποθηκευτούν και να είναι έτοιμα για ανταλλαγή σε ένα δίκτυο blockchain.

Η παρούσα πτυχιακή εργασία δημιουργήθηκε με σκοπό να αναδείξει τις δυνατότητες της τεχνολογίας Blockchain καθώς και τα μειονεκτήματα που υπάρχουν στην εφαρμογή της. Η blockchain εφαρμογή έχει αναπτυχθεί σε ένα τοπικό blockchain δίκτυο και εστιάζει τόσο στην δημιουργία αλλά κα στην διαχείριση των προϊόντων μιας ιστοσελίδας. Κάθε πωλητής θα μπορεί να καταχωρεί τα προϊόντα του και οποιοσδήποτε αγοραστής να τα αγοράζει. Επίσης η εφαρμογή θεωρεί ως δεδομένο ότι θα υπάρχει ένας κρατικός λογαριασμός για την εξασφάλιση του αυτοματισμού είσπραξης και υπολογισμού του φόρου σε κάθε συναλλαγή.

ABSTRACT

The Blockchain technology can be described as a distributed software network. It works as a digital mechanism which allows the safe trade of assets without the need of an intermediate (e.g., bank). Just as web is a technology that facilitates the digital flow of information, so blockchain is a technology that facilitates the digital transaction of items with some value. These items could be anything, from currencies to land that can be stored and exchanged on a blockchain network.

This thesis was created to highlight the capabilities of the blockchain technology and the disadvantages as well that exist in its application. The blockchain application has been developed in a local blockchain network and focuses both on creating and managing the products on a website. Each seller will be able to register his products, while any buyer will be able to buy them. The application also assumes that there will be a state account to ensure the automation of tax collection and calculation in each transaction which takes place on the website.

Πίνακας περιεχομένων

ΚΕΦΑΛΑΙΟ 1 Εισαγωγή

(Υποκεφάλαιο 1.1 Τι είναι το Blockchain)

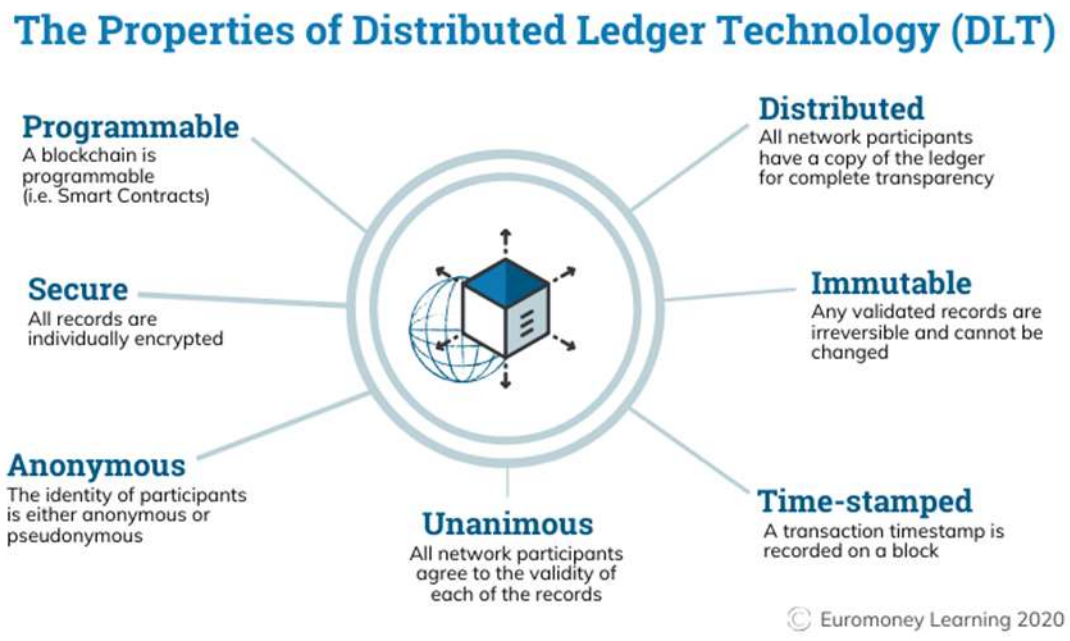
(Ενότητα 1.1.α Ορισμός του Blockchain)

Το Blockchain είναι ένα σύστημα καταγραφής πληροφοριών με τρόπο που καθιστά δύσκολη ή αδύνατη την αλλαγή, την παραβίαση ή την εξαπάτηση του συστήματος.

Ένα blockchain είναι ουσιαστικά ένα ψηφιακό βιβλίο συναλλαγών που αντιγράφεται και διανέμεται σε ολόκληρο το δίκτυο συστημάτων υπολογιστών που είναι στο Blockchain. Κάθε block στην αλυσίδα (blockchain) περιέχει έναν αριθμό συναλλαγών και κάθε φόρα που πραγματοποιείται μια νέα συναλλαγή στην αλυσίδα, μια εγγραφή αυτής της συναλλαγής προστίθεται στο βιβλίο συναλλαγών του κάθε συμμετέχοντα. Η
decentralised (αποκεντρωμένη) βάση δεδομένων που διαχειρίζονται πολλοί συμμετέχοντες είναι γνωστή ως Distributed Ledger Technology (DLT).

Το Blockchain είναι ένας τύπος DLT στο οποίο οι συναλλαγές καταγράφονται με μια αμετάβλητη κρυπτογραφική υπογραφή που ονομάζεται hash (κατακερματισμός). Αυτό σημαίνει, ότι αν ένα block στην αλυσίδα έχει αλλάξει, θα ήταν άμεσα αντιληπτό ότι έχει παραβιαστεί. Εάν οι χάκερ ήθελαν να καταστρέψουν ένα σύστημα blockchain, θα έπρεπε να αλλάξουν κάθε block στην αλυσίδα, σε όλες τις διανεμημένες εκδόσεις της αλυσίδας.

Blockchains σαν το Bitcoin και το Ethereum (Buterin, 2013) συνεχώς αυξάνονται καθώς προστίθενται block στην αλυσίδα. Το γεγονός αυτό συμβάλει σημαντικά στην αύξηση της ασφάλειας του δικτύου. (wha) (Satoshi, 2008)



Εικόνα 1 Τα στοιχεία μιας τεχνολογίας Distributed Ledger

(Ενότητα 1.1 β Το πρόβλημα που προσπαθεί να λύσει το Blockchain)

Έχουν γίνει πολλές προσπάθειες στο παρελθόν για τη δημιουργία ψηφιακού νομίσματος, αλλά πάντα αποτύγχαναν. Το κυρίαρχο ζήτημα, το οποίο δυσκολεύει την δημιουργία κάποιου νομίσματος είναι εμπιστοσύνη. Παραδείγματος χάρι αν κάποιος δημιουργήσει ένα νέο νόμισα, πώς θα μπορούσαμε να τον εμπιστευτούμε ότι δεν θα έδινε στον εαυτό του ένα εκατομμύριο από αυτό το νόμισμα ή δεν θα έκλεβε κάποιου άλλου τα λεφτά για τον εαυτό του.

Το Bitcoin (Satoshi, 2008) σχεδιάστηκε για να λύσει αυτό το πρόβλημα χρησιμοποιώντας έναν συγκεκριμένο τύπο βάσης δεδομένων που ονομάζεται blockchain. Οι περισσότερες κανονικές βάσεις δεδομένων, όπως μια βάση δεδομένων SQL (Jim, 1998), έχουν κάποιον υπεύθυνο που μπορεί να αλλάξει τις καταχωρίσεις (π.χ. να δώσει στον εαυτό του ένα εκατομμύριο δολάρια). Το blockchain είναι διαφορετικό επειδή κανείς δεν είναι υπεύθυνος και διευθύνεται από τους ανθρώπους που το χρησιμοποιούν. Επιπλέον, τα bitcoin δεν μπορούν να παραποιηθούν, να παραβιαστούν ή να δαπανηθούν διπλά. Έτσι όποιος κατέχει αυτά τα νομίσματα μπορεί να εμπιστευτεί ότι έχουν κάποια αξία.

(Υποκεφάλαιο 1.2 Σκοπός της εργασίας)

(Ενότητα 1.2.α Ethereum Blockchain)

Το Ethereum είναι μια πλατφόρμα blockchain που δημιουργεί ένα δίκτυο peer-to-peer που εκτελεί και επαληθεύει τον κώδικα της εφαρμογής, τα οποία ονομάζονται smart contracts (έξυπνα συμβόλαια). (Buterin, 2013)

(Ενότητα 1.2.β Smart Contracts)

Τα smart contracts επιτρέπουν στους συμμετέχοντες να συναλλάζουν μεταξύ τους χωρίς μια αξιόπιστη κεντρική αρχή π.χ. τράπεζα.

(Ενότητα 1.2.γ Το πρόβλημα που θα λύσει η εργασία)

Η παρούσα εργασία θα χρησιμοποιήσει την πλατφόρμα Ethereum και μαζί την κύρια λειτουργία των smart contracts με σκοπό να αναδείξει τα οφέλη της τεχνολογίας σε ένα ηλεκτρονικό κατάστημα.

ΚΕΦΑΛΑΙΟ 2 Βιβλιογραφική Επισκόπηση

(Υποκεφάλαιο 2.1 Επισκοπήσεις)

(Ενότητα 2.1.α A blockchain-based framework of cross-border e-commerce supply chain)

Αυτή η έρευνα εστιάζει στα πλαίσια του διασυνοριακού ηλεκτρονικού εμπορίου, για να προτείνει ένα blockchain-framework. Το οποίο έχει σκοπό να αναπτύξει ένα σύνολο τεχνικών και μεθόδων για την επίτευξη ανιχνεύσιμων προϊόντων και συναλλαγών στη διαχείριση μιας εφοδιαστικής αλυσίδας. Αυτό το framework ενσωματώνει μια σειρά από μοντέλα που βασίζονται σε blockchain, σε δομές πολλαπλών αλυσίδων, σε διαχείριση δεδομένων και δομές block. Επίσης αναπτύσσει αρκετές βασικές μεθόδους και αλγόριθμους. Τα αποτελέσματα της έρευνας δείχνει ότι το framework αυτό έχει την δυνατότητα αντιμετωπίσει οποιοδήποτε πρόβλημα ανάκτησης κλειδιού και να προστατέψει από επιθέσεις κλώνων, επίθεση πλαστών ετικετών και από επιθέσεις απομίμηση προϊόντος (Zhiyong Liu, 2020,).

(Ενότητα 2.1.β Decentralized accessibility of e-commerce products through blockchain technology)

Εξετάζεται ένα κατανεμημένο και διαφανές λογιστικό σύστημα για διάφορα προϊόντα ηλεκτρονικού εμπορίου. Έτσι αυτή η έρευνα παρουσιάζει μια λύση που ονομάζεται «PRODCHAIN». Αυτή η λύση είναι ουσιαστικά ένα γενικό blockchain framework με κρυπτογραφικές διαδικασίες που βασίζονται σε ένα πλέγμα για τη μείωση της πολυπλοκότητας για τον εντοπισμό των προϊόντων ηλεκτρονικού εμπορίου. Η λύση έχει αναλυθεί και έχει πραγματοποιήσει πειραματικές μελέτες στο δίκτυο Ethereum. Τα αποτελέσματα έχουν ως καταλύτη την καθυστέρηση και την απόδοση που αποδεικνύουν την αποτελεσματικότητα του «PRODCHAIN» σε προϊόντα και υπηρεσίες ηλεκτρονικού εμπορίου. Έτσι η λύση είναι επωφελής για τη βελτίωση της ιχνηλασιμότητας των

προϊόντων, διασφαλίζοντας την κοινωνική και οικονομική βιωσιμότητα (Gulshan Kumar, 2020,).

(Ενότητα 2.1.γ Sustainable B2B E-Commerce and Blockchain-Based Supply Chain Finance)

Οι εξελίξεις στην τεχνολογία της πληροφορίας σε συνδυασμό με την αλυσίδα εφοδιασμού ηλεκτρονικού εμπορίου, επιτρέπει στους συμμετέχοντες στο επιχειρησιακό κομμάτι να εργάζονται αποτελεσματικά με μεγάλο όγκο δεδομένων. Επίσης τους επιτρέπει να ελέγχουν τις συναλλαγές. Στα πλαίσια αυτής της έρευνας, για τη βελτίωση της κερδοφορίας και της ανταγωνιστικότητας των εταιριών ηλεκτρονικού εμπορίου, αναλύεται μια blockchain λύση που ενσωματώθηκε στην παγκόσμια αλυσίδα εφοδιασμού B2B (business-to-business). Αυτή η τεχνολογία απλοποίησε τη διαδικασία συναλλαγής παρέχοντας σε όλους τους συμμετέχοντες, στη βιώσιμη διαδικασία αγοράς B2B, τα ίδια δεδομένα σχετικά με το εμπόριο. Η χρήση του blockchain βελτίωσε την αποτελεσματικότητα των logistics και της ψηφιακής τεκμηρίωσης που έφτασε κοντά στο 74% και το 75%, αντίστοιχα. Το κύριο πλεονέκτημα που μας προσφέρει η χρήση του blockchain είναι ότι δημιουργεί μια αποκεντρωμένη βάση δεδομένων που είναι ασφαλής. Επιπλέον, αυξάνει την ταχύτητα πληρωμής, την αξιοπιστία και τη διαφάνεια στη μεταφορά δεδομένων (Lahkani MJ, 2020).

(Ενότητα 2.1.δ Blockchain-based decentralized reputation system in E-commerce environment)

Στο ηλεκτρονικό εμπόριο οι βαθμολογίες φήμης που καταχωρούν οι χρήστες για τα ηλεκτρονικά καταστήματα υπολογίζονται κεντρικά και αποθηκεύονται σε έναν cloud διακομιστή (server). Ο διακομιστής αυτός, μπορεί να κάνει λάθη ή ακόμα και να εμπλέκεται σε απάτη και πλαστογραφία. Για την αντιμετώπιση αυτού του ζητήματος, η έρευνα αυτή προτείνει ένα αποκεντρωμένο σύστημα φήμης που βασίζεται σε blockchain στο περιβάλλον ηλεκτρονικού εμπορίου, δηλαδή τις ηλεκτρονικές αγορές. Σε αυτό το σύστημα οι πληροφορίες προϊόντος, συμπεριλαμβανομένων των περιγραφών και των σχολίων, αποθηκεύονται στο διαπλανητικό σύστημα αρχείων (IPFS) και επιστρέφει την αντίστοιχη διεύθυνση. Οι επιστρεφόμενη διεύθυνση και οι βαθμολογίες φήμης των χρηστών αποθηκεύονται στο blockchain. Η αξιολόγηση της φήμης υλοποιείται με το σχεδιασμό και την ανάπτυξη ενός smart contract στο blockchain. Διαφορετικά από τα παραδοσιακά κεντρικά συστήματα, η προτεινόμενη λύση, μπορεί να προστατεύσει τις πληροφορίες του προϊόντος και τις βαθμολογίες φήμης των χρηστών από σκόπιμες και ακούσιες τροποποιήσεις. Αυτό συμβαίνει διότι είναι αρκετά δύσκολο να αλλάξει οποιαδήποτε δεδομένα είναι αποθηκευμένα στο blockchain και στο IPFS. Η προτεινόμενη λύση προσομοιώνεται στη δημοφιλή πλατφόρμα Ethereum blockchain με τη γλώσσα προγραμματισμού Solidity (Zhili Zhou, 2021).

(Ενότητα 2.1.ε E-commerce Blockchain Consensus Mechanism for Supporting High-Throughput and Real-Time Transaction)

Οι συναλλαγές υπάρχει περίπτωση να έχουν τροποποιηθεί. Αυτό το γεγονός οδηγεί σε χαμηλή αξιοπιστία των συναλλαγών που περιορίζει την ταχεία ανάπτυξη του ηλεκτρονικού εμπορίου. Αν και το blockchain μπορεί να εξασφαλίσει υψηλή σταθερότητα και αξιοπιστία δεδομένων, οι υπάρχουσες blockchain λύσεις εξακολουθούν να έχουν ορισμένα σημαντικά εμπόδια επεκτασιμότητας. Για παράδειγμα, χαμηλή απόδοση και υψηλή καθυστέρηση. Για τη βελτίωση της αξιοπιστίας, αυτή η δημοσίευση παρουσιάζει έναν μηχανισμό συναίνεσης blockchain ηλεκτρονικού εμπορίου (EBCM, e-commerce blockchain consensus mechanism). Αυτός ο μηχανισμός δεν βασίζεται στην υπολογιστική ισχύ και στα tokens, αλλά έχει το ίδιο επίπεδο ασφάλειας και αξιοπιστίας με τη συναίνεση του Nakamoto (Satoshi, 2008). Επιπλέον, επιτυγχάνει συναλλαγές σε πραγματικό χρόνο και υψηλή απόδοση. Με την εισαγωγή του blockchain επικύρωσης, μπορούμε να διασφαλίσουμε ότι οι συναλλαγές δεν μπορούν να αλλάξουν. Προκειμένου να πραγματοποιηθούν συναλλαγές με υψηλή απόδοση και σε πραγματικό χρόνο, αυτή η δημοσίευση κατασκευάζει μια αλυσίδα από block δύο επιπέδων. Ο μηχανισμός έχει συγκριθεί με το Bitcoin ως προς την απόδοση και δείχνει καλύτερη απόδοση.

(Ενότητα 2.1.ζ A Cross-Border E-Commerce Approach Based on Blockchain Technology)

Στο παρόν διασυνοριακού ηλεκτρονικού εμπορίου, οι διάφορες διαδικασίες καταγραφής και εξουσιοδότησης εγγράφων είναι περίπλοκες. Έτσι, η αποτελεσματικότητα κοινής χρήσης αρχείων είναι χαμηλή και η επαλήθευση ταυτότητας δύσκολη. Σε αυτό το δημοσίευμά προτείνεται μια μέθοδος τεχνολογίας ασύμμετρης κρυπτογράφησης που συνδυάζει την τεχνολογία blockchain και την κρυπτογραφία. Με βάση την μη δυνατή τροποποίηση δεδομένων λόγω της φύσης του blockchain και την τεχνολογία ασύμμετρης κρυπτογράφησης, σχεδιάστηκαν συμβάσεις συγχρονισμού αρχείων και συμβάσεις εξουσιοδότησης. Τα πλεονεκτήματα της κατανεμημένης αποθήκευσης διασφαλίζουν το απόρρητο των διασυνοριακών πληροφοριών ηλεκτρονικού εμπορίου των χρηστών. Επιπλέον ο σχεδιασμός της σύμβασης απόκτησης μεταξύ τομέων μπορεί να επαληθεύσει αποτελεσματικά την ταυτότητα και την αποτελεσματικότητα μετάδοσης και των δύο μερών στην κοινή χρήση δεδομένων. Αυτό έχει σκοπό, οι παράνομοι χρήστες να μπορούν να φιλτράρονται με ασφάλεια χωρίς συμβολαιογραφικό ίδρυμα τρίτου μέρους. Τα αποτελέσματα του πειράματος προσομοίωσης δείχνουν ότι, η λύση που προτείνεται σε αυτό το έγγραφο, έχει προφανή πλεονεκτήματα στην αντικλεπτική προστασία δεδομένων και στον έλεγχο ταυτότητας πολλαπλών μερών σε σύγκριση με μια παραδοσιακή αρχιτεκτονική (Hongmei, 2021).

(Ενότητα 2.1.η RTChain: A Reputation System with Transaction and Consensus Incentives for E-commerce Blockchain)

Το Bitcoin είναι η πιο επιτυχημένη εφαρμογή της τεχνολογίας blockchain , όπου επιτρέπει μη απορρίψεις και μη παραποιήσιμες διαδικτυακές συναλλαγές χωρίς τη συμμετοχή ενός έμπιστου κεντρικού μεσάζοντα. Ένας καλά σχεδιασμένος μηχανισμός συναίνεσης, αφενός, πρέπει να είναι αποτελεσματικός για να ανταποκρίνεται στην υψηλή συχνότητα των διαδικτυακών συναλλαγών. Για παράδειγμα, τα υπάρχοντα συστήματα

ηλεκτρονικών πληρωμών μπορούν να χειριστούν πάνω από 50.000 συναλλαγές ανά δευτερόλεπτο (TPS), ενώ το Bitcoin μπορεί να χειριστεί περίπου μόνο 3 TPS κατά μέσο όρο. Σε αυτό το άρθρο καθιερώνεται ένα σύστημα φήμης, που θα ενσωματωθεί στο σύστημα blockchain με σκοπό να επιτύχουμε μια κατανεμημένη συναίνεση συναλλαγών. Το προτεινόμενο σύστημα έχει τα εξής πλεονεκτήματα. Πρώτον, χρησιμοποιείται ένας μηχανισμός κινήτρων για να επηρεάσει τη συναινετική συμπεριφορά των κόμβων και τη συμπεριφορά συναλλαγών των χρηστών. Δεύτερον, το RTChain χρησιμοποιεί μια επαληθεύσιμη τυχαία συνάρτηση για να δημιουργήσει τον «ηγέτη» σε κάθε γύρο. Αυτό εγγυάται το δίκαιο για όλους τους συμμετέχοντες και, σε αντίθεση με το Pow, δεν καταναλώνει μεγάλο όγκο υπολογιστικών πόρων. Τρίτον, δημιουργήθηκε μια αλυσίδα φήμης για την εφαρμογή της κατανεμημένης αποθήκευσης και διαχείριση της φήμης. Τέλος, ο μηχανισμός συναίνεσής είναι ασφαλής έναντι υπαρχουσών επιθέσεων, όπως επιθέσεις flash, επιθέσεις εξόρυξης κλπ. Επίσης επιτρέπει στους κόμβους στη συναίνεσής να αποτύχουν, εφόσον η φήμη του κόμβου αποτυχίας δεν υπερβαίνει το ένα τρίτο της συνολικής φήμης (Sun, 2020).

(Ενότητα 2.1.θ Smart contracts for blockchain-based reputation systems: A systematic literature review)

Αυτό το άρθρο, μέσω μιας συστηματικής βιβλιογραφικής ανασκόπησης, διαπιστώνει ότι η υπάρχουσα βιβλιογραφία δεν έχει προτείνει ένα πλαίσιο που να διευκολύνει την εναλλάξιμη χρήση smart contracts για συστήματα φήμης blockchain. Έτσι προτείνεται ένα framework FarMed για τη δημιουργία ενός ευφυούς πλαισίου. Αυτό το πλαίσιο θα εκτελεί συστήματα φήμης που βασίζονται σε έξυπνες επαφές Ethereum και θα αναπτύσσει αξιόπιστα πρωτόκολλα βασισμένα στο blockchain για τη μεταφορά τιμών φήμης από τον έναν πάροχο στον άλλον (Ahmed S. Almasoud, 2020).

(Ενότητα 2.1.ι Blockchain-Enabled Smart Contracts: Architecture, Applications, and Future Trends)

Σε αυτό το δημοσίευμα, παρουσιάστηκε μια συστηματική και περιεκτική επισκόπηση των smart contracts της τεχνολογίας blockchain με στόχο την τόνωση περαιτέρω έρευνας σε αυτόν τον αναδυόμενο ερευνητικό τομέα. Πρώτον, παρουσιάζεται ο μηχανισμός λειτουργίας και οι κύριες πλατφόρμες των smart contracts. Επιπλέον, προτείνεται ένα ερευνητικό πλαίσιο για έξυπνες συμβάσεις που βασίζεται σε μια αρχιτεκτονική έξι επιπέδων. Δεύτερον, παρατίθενται τόσο οι τεχνικές και νομικές προκλήσεις, όσο και η πρόσφατη πρόοδος της ερευνάς αυτής. Τέλος, παρουσιάζονται αρκετά τυπικά σενάρια εφαρμογής της έρευνας (S. Wang, 2019).

(Ενότητα 2.1.κ Ethereum Smart Contracts as Blockchain-oriented Microservices)

Σε αυτό το δημοσίευμα προτείνεται ένα μοντέλο αρχιτεκτονικής λογισμικού όπου τα microservices υλοποιούνται μέσω smart contracts που αναπτύσσονται σε ένα blockchain. Επίσης, συζητά τις ομοιότητες μεταξύ των δύο παραδειγμάτων και παρουσιάζει ένα παράδειγμα εφαρμογής μιας πλατφόρμας ηλεκτρονικού εμπορίου (Roberto Tonelli, 2018).

(Ενότητα 2.1.λ Blockchain-enabled logistics finance execution platform for capital-constrained E-commerce retail)

Αυτή η δημοσίευση προτείνει μια blockchain πλατφόρμα χρηματοδότησης διοικητικής μέριμνας ως μια ολοκληρωμένη λύση για τη διευκόλυνση του LF (logistics finance) για το ηλεκτρονικό εμπόριο λιανικής. Έτσι, προτείνεται μια αρχιτεκτονική πολλαπλών επιπέδων για την οργάνωση και τη διαχείριση εμπλεκόμενων πόρων, ροών εργασίας και αποφάσεων με βάση την αντικειμενοστραφή μεθοδολογία (OOM). Ένα υβριδικό έξυπνο συμβόλαιο που βασίζεται σε μηχανές πεπερασμένης κατάστασης (HFSM-SC) έχει σχεδιαστεί για να συσχετίζεται και να συντονίζεται με όλα τα είδη πρακτόρων για λειτουργίες LF καθ' όλη τη διάρκεια του κύκλου ζωής του. Τέλος, διεξάχθηκε μια μελέτη για την εφαρμογή δυναμικής διαχείρισης δεσμεύσεων με δυνατότητα BcLFEP για την επαλήθευση και την αξιολόγηση της έρευνας (Ming Li, 2020).

(Ενότητα 2.1.μ Log-flock: A blockchain-enabled platform for digital asset valuation and risk assessment in E-commerce logistics financing)

Αυτή η μελέτη εισάγει μια πλατφόρμα χρηματοδότησης διοικητικής μέριμνας (Log-Flock), που αξιοποιεί τις ικανότητες του Internet Of Things(IoT), του cyber-physical system (CPS) και των τεχνολογιών blockchain για την υποστήριξη της χρήσης των ψηφιακών περιουσιακών στοιχείων του LC για τη χρηματοδότηση διοικητικής μέριμνας. Το IoT και το CPS συνδυάζονται για να υποστηρίξουν τη δημιουργία των ψηφιακών στοιχείων, ενώ οι βασικές δομές της τεχνολογίας blockchain τα tokens και τα smart contracts (Arjun Rachana Harish, 2021).

(Ενότητα 2.1.ν Incentive-Driven Computation Offloading in Blockchain-Enabled E-Commerce)

Το blockchain θεωρείται μια από τις πιο υποσχόμενες τεχνολογίες για την αναβάθμιση του ηλεκτρονικού εμπορίου. Αυτή η δημοσίευση αναλύει τις προκλήσεις που αντιμετωπίζει το τρέχον ηλεκτρονικό εμπόριο και εισάγει ένα νέο σενάριο που ενεργοποιείται μέσα από το blockchain. Προτείνεται ένα framework για εργασίες εξόρυξης σε αυτό το σενάριο που εκφορτώνονται σε διακομιστές αιχμής και βασίζονται σε υπολογιστές ακμών για φορητές συσκευές. Στην συνέχεια, το ζήτημα της εκφόρτωσης μοντελοποιείται ως ένα πρόβλημα βελτιστοποίησης πολλαπλών περιορισμών. Στο πρόβλημα αυτό χρησιμοποιούνται και επανασχεδιάζονται ως λύτες εξελικτικοί αλγόριθμοι (Shuiguang Deng, 2020).

(Ενότητα 2.1.ξ Blockchain Technology for E-commerce Industry)

Το δημοσίευμα αυτό εξέτασε την τεχνολογία blockchain στον κλάδο του ηλεκτρονικού εμπορίου. Εφαρμογές blockchain συζητούνται για διάφορες πτυχές του ηλεκτρονικού εμπορίου όπως, πληρωμή, ασφάλεια, εφοδιαστική αλυσίδα, αυτοματισμός εργασίας με smart contracts, ηθικές πρακτικές για διαφάνεια σε συναλλαγές ηλεκτρονικού εμπορίου (Hemantkumar P. Bulsara, 2020).

(Ενότητα 2.1.ο ETTF: A Trusted Trading Framework Using Blockchain in E-commerce)

Αυτό το δημοσίευμα παρουσιάζει ένα αξιόπιστο framework (ETT F) που χρησιμοποιεί το πρωτόκολλο blockchain στο ηλεκτρονικό εμπόριο για την επίτευξη μιας περισσότερο αξιόπιστης συναλλαγής. Το ETTF περιλαμβάνει ένα πρωτόκολλο ομότιμων blockchain (PBP) που βασίζεται σε μια αρχιτεκτονική για την υποστήριξη της αποθήκευσης μαζικών και άμεσων συναλλαγών. Στο PBP, οι κλίμακες απόδοσης αυξάνονται σχεδόν γραμμικά με τον υπολογισμό: όσο περισσότερη υπολογιστική ισχύς είναι διαθέσιμη, τόσο περισσότερα block επιλέγονται ανά μονάδα χρόνου. Επιπλέον, προκειμένου να διασφαλίσουμε μεγαλύτερη ασφάλεια των συναλλαγών, έχει εισαχθεί ένας ισχυρός αλγόριθμος συναίνεσης (ECA) στο ηλεκτρονικό εμπόριο. Σε σύγκριση με το Bitcoin blockchain, το ETTF δείχνει καλύτερη απόδοση στους τομείς που αφορά την καθυστέρηση και τη χωρητικότητα στο ηλεκτρονικό εμπόριο ({Xie, 2018).

ΚΕΦΑΛΑΙΟ 3 Blockchain

(Υποκεφάλαιο 3.1 Η δομή ενός Blockchain)

(Ενότητα 3.1.α Αρχιτεκτονική)

Ένα Blockchain συνήθως αποτελείται από 6 βασικά αρχιτεκτονικά στοιχεία :

(1) Κόμβος (Node), το οποίο είναι είτε ένας χρήστης είτε ένας υπολογιστής εντός του blockchain και ο καθένας διατηρεί ένα ανεξάρτητο αντίγραφο από όλο το blockchain σύστημα.

(2) Συναλλαγή (transaction), το μικρότερο δομικό στοιχείο ενός συστήματος blockchain όπου περιλαμβάνει αρχεία, πληροφορίες κ.λπ. και χρησιμεύει ως σκοπός του blockchain.

(3) Block, το οποίο είναι μια δομή δεδομένων για τη διατήρηση ενός συνόλου συναλλαγών που διανέμεται σε όλους τους κόμβους του δικτύου.

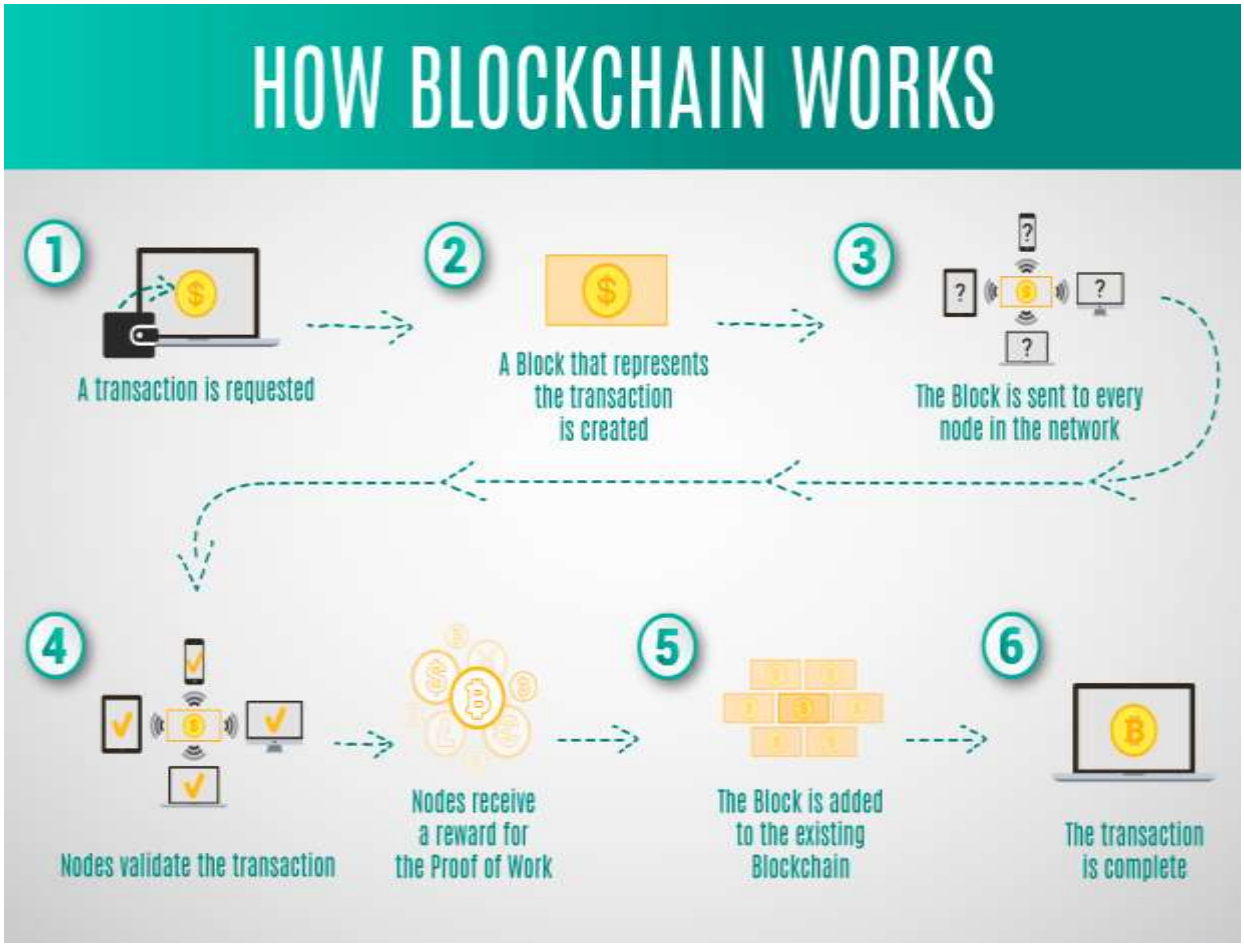
(4) Αλυσίδα (chain), μια ακολουθία από blocks σε μια συγκεκριμένη σειρά.

(5) Εξορύκτες (miners), συγκεκριμένοι κόμβοι που πραγματοποιούν την διαδικασία επαλήθευσης ενός block προτού προστεθεί οτιδήποτε πάνω στο σύστημα του blockchain

(6) Πρωτόκολλο συναίνεσης (Consensus protocol), ένα σύνολο κανόνων και ρυθμίσεων για την εκτέλεση λειτουργιών του blockchain.

Όταν οποιαδήποτε νέα συναλλαγή ή εγγραφή πραγματοποιείται, συνεπάγεται η δημιουργία ενός νέου block. Κάθε εγγραφή στη συνέχεια αποδεικνύεται από την μέθοδο proof-of-work(Pow) και υπογράφεται ψηφιακά για να διασφαλιστεί η γνησιότητα και η αυθεντικότητά του. Πριν προστεθεί στο δίκτυο αυτό το νέο block, θα πρέπει να επαληθευτεί από την πλειοψηφία των κόμβων του συστήματος.

Το παρακάτω διάγραμμα αρχιτεκτονικής blockchain δείχνει πως λειτουργεί στην πραγματικότητα με τη μορφή ψηφιακού πορτοφολιού.



Εικόνα 2 Πως λειτουργεί το blockchain

(Ενότητα 3.1.β Η δομή ενός Block)

Σε κάθε blockchain, η δομή του block αποτελείται από 3 στοιχεία:

1. Κάποια δεδομένα
2. Το hash του block
3. Το hash του προηγούμενου block

Τα δεδομένα που αποθηκεύονται σε κάθε μπλοκ εξαρτώνται από τον τύπο του blockchain. Για παράδειγμα στην αρχιτεκτονική του Bitcoin blockchain, το block διατηρεί δεδομένα σχετικά με τον αποστολέα - παραλήπτη αλλά και την ποσότητα των νομισμάτων.

Ένα hash είναι σαν ένα δακτυλικό αποτύπωμα (μεγάλη εγγραφή που αποτελείται από μερικά ψηφία και γράμματα). Κάθε hash παράγεται με την βοήθεια του κρυπτογραφικού αλγόριθμου SHA 256. Συνεπώς, αυτό βοηθάει στην αναγνώριση του κάθε block σε μια

blockchain δομή . Την στιγμή που δημιουργείται ένα block, αυτομάτως του επισυνάπτεται ένα hash και οποιαδήποτε αλλαγή πραγματοποιηθεί στο block επηρεάζει και την αλλαγή του hash. Στην ουσία τα hashes βοηθούν στον εντοπισμό τυχόν αλλαγών στα blocks.

Κάθε block περιέχει και το hash του προηγούμενου block της αλυσίδας. Αυτό δημιουργεί μια αλυσίδα από blocks και είναι το κύριο στοιχείο πίσω από την αρχιτεκτονική της ασφάλειας του blockchain. Το πρώτο block σε μια αλυσίδα είναι λίγο ιδιαίτερο, καθώς όλα τα επιβεβαιωμένα και επικυρωμένα blocks προέρχονται από αυτό. Το πρώτο block ονομάζεται Genesis Block.

Οποιαδήποτε κακόβουλη προσπάθεια πραγματοποιηθεί, προκαλεί τα blocks να αλλάξουν. Στη συνέχεια, όλα τα ακόλουθα blocks φέρουν εσφαλμένες πληροφορίες και ολόκληρο το blockchain σύστημα καθίσταται μη έγκυρο. (Buterin, 2013)

(Ενότητα 3.1.γ Genesis Block)

Όπως αναφέρθηκε προηγουμένως το πρώτο block σε μια αλυσίδα ονομάζεται Genesis Block ή αλλιώς και Block 0. Αποτελεί τη βάση του συστήματος συναλλαγών και είναι το πρωτότυπο όλων των άλλων block στην αλυσίδα. Είναι ουσιαστικά ο πρόγονος στον οποίο κάθε άλλο block μπορεί να ανιχνεύσει τη γενεαλογία του, αφού κάθε block διατηρεί μία αναφορά (ένα hash πιο συγκεκριμένα) του προηγούμενού του. Ακόμη, είναι σχεδόν πάντα hardcoded (κωδικοποιημένο) μέσα στο λογισμικό των εφαρμογών που χρησιμοποιούν το blockchain. Είναι μια ειδική περίπτωση καθώς δεν περιέχει αναφορά σε προηγούμενο block και επίσης στην περίπτωση του Bitcoin, και σχεδόν σε όλα τα παράγωγα του, παράγει μια μη αναλώσιμη επιδότηση (subsidy).

```
GetHash()     = 0x000000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f
hashMerkleRoot = 0x4a5e1e4baab89f3a32518a88c31bc87f618f76673e2cc77ab2127b7afdeda33b
txNew.vin[0].scriptSig     = 486604799 4 0x736B6E6162207 26F6620747 56F6C6961 2206 46E6F6365732066 6F 20686E6972 2206E6F 20726F6C6C65636E6168432039303032 2F6E614A2F33302073656D6954206 56854
txNew.vout[0].nValue       = 5000000000
txNew.vout[0].scriptPubKey = 0x5F1DF16B2B704C8A578D0BBAF740385CDE12C11EE50455F3C438EF4C3FBCF649B6DE611FEAE06279A60939E028A8D65C10B73071A6F16719274855FEB0FD846704 OP_CHECKSIG
block.nVersion = 1
block.nTime    = 1231006505
block.nBits    = 0x1d00ffff
block.nNonce   = 2083236893

CBlock(hash=000000000019d6, ver=1, hashPrevBlock=0000000000000, hashMerkleRoot=4a5e1e, nTime=1231006505, nBits=1d00ffff, nNonce=2083236893, vtx=1)
  CTransaction(hash=4a5e1e, ver=1, vin.size=1, vout.size=1, nLockTime=0)
    CTxIn(COutPoint(000000, -1), coinbase 04ffff001d0104554686520546963573030332f4a616e2f323030392043686616e63656c6c6f72206f6e20627696e6b206f662073656368616e63652061696c6f757420666f722062616e6b73)
    CTxOut(nValue=50.00000000, scriptPubKey=0x5F1DF16B2B704C8A578D0B)
  vMerkleTree: 4a5e1e
```

Εικόνα 3 Bitcoin Genesis Block

Η εικόνα 3 αναπαριστά το block genesis όπως είχε εμφανιστεί σε ένα σχόλιο σε μια παλιά έκδοση του Bitcoin. Η πρώτη ενότητα ορίζει ακριβώς όλες τις απαραίτητες μεταβλητές για την αναδημιουργία του block. Η δεύτερη ενότητα είναι το block σε τυπική μορφή printblock (μπλοκ εκτύπωσης), το οποίο περιέχει την σύντομη έκδοση δεδομένων από την πρώτη ενότητα.

Το hash του genesis block έχει δυο περισσότερα hex (εξάγωνα) μηδενικά στην αρχή του, από ότι απαιτείται για ένα πρώιμο block. (Genesis block, n.d.)

(Ενότητα 3.1.δ Proof of work)

Στην θεωρία, θα μπορούσε να ήταν δυνατή η προσαρμογή όλων των block με την βοήθεια ισχυρών επεξεργαστών. Ωστόσο, υπάρχει μια λύση που εξαλείφει αυτή τη δυνατότητα που ονομάζεται proof-of-work (απόδειξη εργασίας). Αυτό επιτρέπει στον χρήστη να επιβραδύνει την διαδικασία δημιουργίας νέων block. Στην αρχιτεκτονική του Bitcoin blockchain, χρειάζεται περίπου 10 λεπτά για να προσδιοριστεί το απαραίτητο proof-of-work και να προστεθεί ένα νέο block στην αλυσίδα. Αυτή η διαδικασία πραγματοποιείται από τους Miners, όπου είναι ειδική κόμβοι (nodes) εντός της δομής Bitcoin blockchain. Οι miners μπορούν να κρατήσουν τα τέλη συναλλαγής από το block που επαλήθευσαν ως ανταμοιβή.

(Ενότητα 3.1.ε Κόμβος)

Ένα blockchain βασίζεται σε blocks δεδομένων. Αυτά τα blocks αποθηκεύονται σε κόμβους. Ένας κόμβος μπορεί να είναι κάθε είδους συσκευή (κυρίως υπολογιστές, φορητοί υπολογιστές ή ακόμα μεγάλοι servers (διακομιστές)). Οι κόμβοι αποτελούν την υποδομή ενός blockchain. Όλοι οι κόμβοι σε ένα δίκτυο συνδέονται μεταξύ τους καθώς επίσης ανταλλάσσουν συνεχώς τα πιο πρόσφατα δεδομένα. Έτσι όλοι οι κόμβοι μπορούν να παραμένουν ενημερωμένοι. Αποθηκεύουν, διαδίδουν και διατηρούν τα δεδομένα του blockchain, επομένως θεωρητικά το blockchain υπάρχει πάνω στους κόμβους.

Κάθε νέος χρήστης (κόμβος) που εντάσσεται στο δίκτυο του blockchain λαμβάνει ένα πλήρες αντίγραφο του συστήματος. Μόλις δημιουργηθεί ένα νέο block, αποστέλλεται σε κάθε κόμβο εντός του συστήματος blockchain. Στη συνέχεια, κάθε κόμβος επαληθεύει το block και ελέγχει εάν οι πληροφορίες που αναφέρονται είναι σωστές. Εάν είναι όλα εντάξει, το block προστίθεται στο τοπικό blockchain του κάθε κόμβου.

(Ενότητα 3.1.ζ Consensus Protocol)

Όλοι οι κόμβοι σε μια αρχιτεκτονική blockchain δημιουργούν ένα consensus protocol (πρωτόκολλο συναίνεσης). Ένα σύστημα συναίνεσης αποτελείται από ένα σύνολο κανόνων δικτύου. Εάν όλοι οι κόμβοι τηρούν αυτούς τους κανόνες, τότε γίνονται αυτοεπιβεβλημένοι (self-enforced) μέσα στο blockchain.

Ωστόσο, η διαδικασία σχεδίασης ενός κατάλληλου πρωτόκολλου είναι ένα μεγάλο ζήτημα. Σε μια παραδοσιακή αρχιτεκτονική λογισμικού, η συναίνεση δεν αποτελεί

πρόβλημα σχεδιασμού, λόγω της ύπαρξης ενός κεντρικού server (διακομιστή). Κατά συνέπεια οι άλλοι κόμβοι χρειάζεται μόνο να ευθυγραμμιστούν με τον κεντρικό server. Ωστόσο, σε ένα κατανεμημένο δίκτυο όπως το blockchain, κάθε κόμβος είναι ταυτόχρονα ένας κεντρικός υπολογιστής αλλά και ένας server, όπου χρειάζεται να ανταλλάσσει πληροφορίες με άλλους κόμβους για να επιτευχθεί η συναίνεση.

Για παράδειγμα, το Bitcoin blockchain έχει έναν κανόνα συναίνεσης ο οποίος δηλώνει ότι ένα ποσό συναλλαγής πρέπει να μειώνεται στο μισό, ανά 200.000 blocks. Τεχνικά, αυτό σημαίνει ότι εάν ένα block παράγει μια ανταμοιβή επαλήθευσης 10 BTC, αυτή η τιμή πρέπει να μειώνεται στο μισό κάθε φορά μετά από 200.000 blocks.

(Υποκεφάλαιο 3.2 Ethereum Blockchain)

(Ενότητα 3.2.α Εισαγωγή στο Ethereum)

Ο σκοπός του Ethereum είναι να δημιουργήσει ένα πρωτόκολλο για τη δημιουργία decentralized applications (αποκεντρωμένων εφαρμογών). Για να συμβεί αυτό το Ethereum παρέχει ένα σύνολο αντισταθμίσεων που πιστεύει ότι θα είναι πολύ χρήσιμες για μια μεγάλη κατηγορία DApps (decentralized applications). Δίνει ιδιαίτερη έμφαση σε καταστάσεις στις οποίες ο γρήγορος χρόνος ανάπτυξης, η ασφάλεια για μικρές και σπάνια χρησιμοποιημένες εφαρμογές αλλά και η ικανότητα διαφορετικών εφαρμογών να αλληλεπιδρούν αποτελεσματικά, είναι σημαντικά. Το Ethereum το επιτυγχάνει αυτό χτίζοντας ουσιαστικά ένα αφηρημένο θεμελιώδης στρώμα. Το στρώμα αυτό είναι ένα blockchain με μια ενσωματωμένη γλώσσα προγραμματισμού Turing, η οποία επιτρέπει σε οποιονδήποτε να γράφει smart contracts (έξυπνα συμβόλαια) καθώς και DApps εφαρμογές. Σε αυτά μπορεί να δημιουργήσει τους δικούς του αυθαίρετους κανόνες ιδιοκτησίας, μορφές συναλλαγών και συναρτήσεις μετάβασης κατάστασης. (Buterin, 2013)

(Ενότητα 3.2β Λογαριασμοί Ethereum)

Στο Ethereum blockchain, η κατάσταση (state) αποτελείται από αντικείμενα που ονομάζονται λογαριασμοί (accounts). Κάθε λογαριασμός περιέχει: μια διεύθυνση 20 byte, καταστάσεις μεταβάσεων, οι οποίες είναι οι άμεσες μεταφορές αξίας και πληροφοριών μεταξύ λογαριασμών, το nonce, δηλαδή έναν μετρητή που χρησιμοποιείται για να βεβαιωθεί ότι κάθε συναλλαγή μπορεί να διεκπεραιωθεί μόνο μια φορά, το τρέχον υπόλοιπο ether του λογαριασμού, τον κωδικό σύμβασης του λογαριασμού, εάν υπάρχει, και τέλος τον αποθηκευτικό χώρο του λογαριασμού.

Γενικά, υπάρχουν δύο τύποι λογαριασμών: α) λογαριασμοί εξωτερικού τύπου ιδιοκτησίας, που ελέγχονται από ιδιωτικά κλειδιά και β) λογαριασμοί συμβολαίου, που ελέγχονται από τον κώδικα συμβολαίου τους. Ένας εξωτερικός λογαριασμός δεν περιέχει κωδικό, και για να στείλει κάποιος ένα μήνυμα χρειάζεται να δημιουργήσει και να υπογράψει μια συναλλαγή, σε έναν λογαριασμό συμβολαίου. Κάθε φορά που ο λογαριασμός συμβολαίου λαμβάνει ένα μήνυμα ο κωδικός ενεργοποιείται, επιτρέποντας του να διαβάσει και να γράψει στον εσωτερικό χώρο αποθήκευσης. Ακόμη μπορεί να στέλνει άλλα μηνύματα ή να δημιουργεί συμβόλαια με την σειρά του. (Buterin, 2013)

(Ενότητα 3.2.γ Ether)

Ether (ETH) είναι το εγγενές κρυπτονόμισμα του Ethereum blockchain. Ο σκοπός του ether είναι να επιτρέψει σε μια αγορά τους υπολογισμούς. Μια τέτοια αγορά παρέχει ένα οικονομικό κίνητρο στους συμμετέχοντες να επαληθεύουν, να εκτελούν αιτήματα συναλλαγών και να παρέχουν υπολογιστικούς πόρους στο δίκτυο. Κάθε συμμετέχων που εκπέμπει ένα αίτημα συναλλαγής πρέπει επίσης να προσφέρει κάποια ποσότητα ether στο δίκτυο ως δώρο (Bounty). Αυτό το bounty θα απονεμηθεί σε όποιον τελικά επαληθεύσει την συναλλαγή, την εκτελέσει, την δεσμεύσει στο blockchain και την μεταδώσει στο δίκτυο.

Το ether ποσό που καταβάλλεται αντιστοιχεί στον χρόνο που απαιτείται για να γίνει ο υπολογισμός. Αυτά τα bounties εμποδίζουν επίσης κακόβουλους συμμετέχοντες από το να φράζουν σκόπιμα το δίκτυο ζητώντας την εκτέλεση άπειρων υπολογισμών ή άλλων σεναρίων έντασης πόρων, καθώς οι συμμετέχοντες πρέπει να πληρώσουν για τον χρόνο υπολογισμού. (what is ether, n.d.)

(Ενότητα 3.2δ Smart Contracts)

Οι συμμετέχοντες δεν γράφουν νέο κώδικα κάθε φορά που θέλουν να ζητήσουν έναν υπολογισμό από το EVM (Ethereum Virtual Machine). Αντίθετα, οι προγραμματιστές εφαρμογών ανεβάζουν προγράμματα (επαναχρησιμοποιήσιμα αποσπάσματα κώδικα) στον χώρο αποθήκευσης του EVM και οι χρήστες υποβάλλουν αιτήματα για την εκτέλεση αυτών των αποσπασμάτων κώδικα με ποικίλες παραμέτρους. Τα προγράμματα που ανεβαίνουν και εκτελούνται από το δίκτυο ονομάζονται smart contracts (έξυπνα συμβόλαια).

Σε μια πιο ελεύθερη μετάφραση, μπορούμε να παρομοιάσουμε τα smart contracts σαν ενός είδους μηχάνημα αυτόματης πώλησης, δηλαδή ένα script που όταν καλείται με συγκεκριμένους παραμέτρους, εκτελεί ορισμένες ενέργειες ή υπολογισμούς εάν πληρούνται ορισμένες προϋποθέσεις. Για παράδειγμα, ένα απλό smart contract θα μπορούσε να δημιουργήσει και να εκχωρήσει την ιδιοκτησία ενός ψηφιακού στοιχείου εάν αυτός που το κάλεσε στείλει ether σε έναν συγκεκριμένο παραλήπτη. Οποιοσδήποτε προγραμματιστής μπορεί να δημιουργήσει ένα smart contract και να το δημοσιεύσει στο δίκτυο, και να χρησιμοποιήσει το blockchain σαν την βάση δεδομένων του για μία αμοιβή που θα καταβάλλεται στο δίκτυο. Στην συνέχεια, οποιοσδήποτε χρήστης μπορεί να καλέσει το smart contract για να εκτελεστεί ο κώδικάς του και πάλι έναντι χρέωσης που καταβάλλεται στο δίκτυο.

Συνεπώς, με την βοήθεια των smart contracts οι προγραμματιστές μπορούν να δημιουργήσουν και να δημοσιεύσουν στο δίκτυο αυθαίρετες πολύπλοκες εφαρμογές που έχουν στόχο τον χρήστη, όπως χρηματοοικονομικά μέσα, παιχνίδια κ.λπ. (what are smart contracts, n.d.)

(Ενότητα 3.2ε Bitcoin vs Ethereum)

Στην πραγματικότητα ακόμη και χωρίς επεκτάσεις, το πρωτόκολλο Bitcoin παρέχει μια αδύναμη εκδοχή των smart contracts. Το UTXO (ποσό ψηφιακού νομίσματος που έχει απομείνει σε κάποιον μετά την εκτέλεση μιας συναλλαγής κρυπτονομίσματος) στο Bitcoin μπορεί να ανήκει όχι μόνο σε ένα δημόσιο κλειδί, αλλά και σε ένα πιο περίπλοκο script που εκφράζεται σε μια απλή γλώσσα προγραμματισμού που βασίζεται στη στοίβα. Σε αυτό το παράδειγμα, μια δαπάνη συναλλαγής το UTXO πρέπει να παρέχει δεδομένα που να ικανοποιούν το script. Ακόμη και ο βασικός μηχανισμός ιδιοκτησίας δημόσιου κλειδιού υλοποιείται μέσω ενός script.

Παρόλα αυτά, η scripting γλώσσα προγραμματισμού που έχει υλοποιηθεί στο Bitcoin έχει αρκετά σημαντικούς περιορισμούς, όπως έλλειψη Turing-completeness, δηλαδή ενώ υπάρχει μεγάλο υποσύνολο υπολογισμών που υποστηρίζει η Bitcoin scripting γλώσσα δεν υποστηρίζει τα πάντα. Η κύρια λειτουργία που λείπει είναι οι βρόγχοι. Παρόλα αυτά η έλλειψή αυτού του στοιχείου είναι σκόπιμη καθώς γίνεται για να αποφευχθούν άπειροι βρόχοι κατά τη επαλήθευση μιας συναλλαγής. Επίσης δεν υπάρχει τρόπος για ένα UTXO script να παρέχει λεπτομερή έλεγχο του ποσού που μπορεί να αποσυρθεί και αυτό ονομάζεται value-blindness (τιμή-τυφλότητας). Ακόμα, υπάρχει έλλειψη κατάστασης (lack of state), δηλαδή το UTXO μπορεί είτε να δαπανηθεί, είτε να μην δαπανηθεί. Δεν υπάρχει καμία δυνατότητα για συμβόλαια (smart contracts) πολλαπλών σταδίων ή scripts που διατηρούν οποιαδήποτε άλλη εσωτερική κατάσταση. Αυτό καθιστά δύσκολη τη σύναψη contracts πολλαπλών σταδίων, decentralized προσφορών ανταλλαγής ή κρυπτογραφικών πρωτοκόλλων δέσμευσης δύο σταδίων (απαραίτητα για ασφαλείς υπολογιστικές αμοιβές). Σημαίνει επίσης, ότι το UTXO μπορεί να χρησιμοποιηθεί μόνο για τη δημιουργία απλών one-off (εφάπαξ) contracts και όχι πιο περίπλοκων stateful (κρατικών) συμβάσεων, όπως decentralized οργανισμοί. Έτσι καθιστά δύσκολη την υλοποιήσει των μετα-πρωτοκόλλων (meta-protocols). Η δυαδική κατάσταση σε συνδυασμό με value-blindness σημαίνει επίσης ότι μια ακόμη σημαντική εφαρμογή όπως τα όρια ανάληψης είναι αδύνατη. Επιπλέον τα UTXO είναι τυφλά σε δεδομένα blockchain, όπως το nonce, το timestamp και το προηγούμενο block hash. Αυτό έχει ως συνέπεια να περιορίζει σοβαρά τις εφαρμογές στον τζόγο, και σε πολλές άλλες κατηγορίες, στερώντας τη γλώσσα από μια δυνητικά πολύτιμη πηγή τυχαίας πιθανότητας.

Ο κώδικας στα Ethereum contracts είναι γραμμένος σε μια γλώσσα χαμηλού επιπέδου bytecode, βασισμένη στη στοίβα και αναφέρεται ως Ethereum virtual machine code ή EVM. Ο κώδικας αποτελείται από μία σειρά byte, όπου το καθένα αντιπροσωπεύει μια λειτουργία. Γενικότερα, η εκτέλεση κώδικα είναι ένας άπειρος βρόχος που αποτελείται από την επανειλημμένη εκτέλεση της λειτουργίας του στον τρέχοντα μετρητή προγράμματος (το οποίο ξεκινά από το μηδέν) και στη συνέχεια αυξάνει τον μετρητή του προγράμματος κατά ένα, μέχρι να φτάσει στο τέλος του κώδικα ή σε ένα σφάλμα ή STOP ή ανιχνευτεί η εντολή RETURN. Οι λειτουργίες έχουν πρόσβαση σε τρείς τύπους χώρου

στους οποίους αποθηκεύονται δεδομένα. Την στοίβα, το οποίο είναι ένα κοντέινερ last-in-first-out στο οποίο μπορούν να ωθηθούν και να εμφανιστούν οι τιμές. Την μνήμη, η οποία είναι ένας άπειρα επεκτάσιμος πίνακας από bytes, και την αποθήκη (storage) όπου είναι ένα μακροπρόθεσμο μέσο αποθήκευσης του contract του οποίου το μοντέλο δεδομένων βασίζεται στο ότι κάθε κλειδί συσχετίζεται με μία και μόνο τιμή. Αντιθέτως με την στοίβα και την μνήμη, τα οποία επαναφέρονται μετά το τέλος του υπολογισμού, στην αποθήκη παραμένουν μακροπρόθεσμα τα δεδομένα. Ο κώδικας μπορεί επίσης να έχει πρόσβαση στην τιμή, στον αποστολέα και τα δεδομένα του εισερχόμενου μηνύματος, στα δεδομένα κεφαλίδας του block και επίσης ο κώδικας μπορεί να επιστρέψει έναν πίνακα από bytes ως δεδομένα εξόδου. (Buterin, 2013)

ΚΕΦΑΛΑΙΟ 4 Σχεδιασμός

(Υποκεφάλαιο 4.1 Blockchain solutions)

(Ενότητα 4.1.α A Transparent Supply Chain)

Το blockchain θα φέρει πιθανώς μεγάλες αλλαγές στον οικονομικό τομέα, όμως ένας άλλος τομέας στον οποίο μπορεί να φέρει μεγάλες αλλαγές είναι στη διαχείριση της εφοδιαστικής αλυσίδας. Η τεχνολογία του blockchain μπορεί να βελτιώσει σημαντικά της αλυσίδες εφοδιασμού επιτρέποντας την ταχύτερη και πιο οικονομική παράδοση προϊόντων, τη βελτίωση στην ιχνηλασιμότητα των προϊόντων, τη βελτίωση στον συντονισμό μεταξύ των εταιριών αλλά και βοηθώντας την πρόσβαση στην χρηματοδότηση.

Γενικότερα, έχει σημειωθεί σημαντική πρόοδος στην ανταλλαγή πληροφοριών στην αλυσίδα εφοδιασμού από τη δεκαετία του 1990, χάρη στην χρήση συστημάτων ERP (προγραμματισμού πόρων επιχείρησης) (Enterprise resource planning, n.d.). Ωστόσο, η διαφάνεια παραμένει μια πρόκληση σε μεγάλες αλυσίδες εφοδιασμού που περιλαμβάνουν πολύπλοκες συναλλαγές. Τα ERP συστήματα που χρησιμοποιούνται τώρα δεν επιτρέπουν στα τρία μέρη που εμπλέκονται σε μια απλή συναλλαγή να βλέπουν όλες τις σχετικές ροές πληροφοριών, αποθέματος και χρημάτων. Το σύστημα blockchain μπορεί να εξαλείψει αυτά τα τυφλά σημεία.

Ένα συχνός τρόπος για τη βελτίωση της εφοδιαστικής αλυσίδας είναι η επαλήθευση συναλλαγών μέσω ελέγχων. Ο έλεγχος είναι απαραίτητος για τη διασφάλιση της συμμόρφωσης με τις συμβάσεις. Από την άλλη όμως, παρέχει περιορισμένη βοήθεια στη βελτίωση της λήψης αποφάσεων για την αντιμετώπιση των λειτουργικών ελλείψεων. Ωστόσο, ένα αρχείο που παρέχει τον έλεγχο σε οποιοδήποτε μέρος της αλυσίδας εφοδιασμού θα μπορέσει να βοηθήσει σημαντικά.

Έτσι, το blockchain μπορεί να μειώσει αρκετά τα προβλήματα της ανιχνευσιμότητας αλλά και του συντονισμού, από την στιγμή που ο κάθε χρήστης έχει το δικό του αντίγραφο του blockchain και μπορεί να ελέγξει την κατάσταση μιας συναλλαγής, να εντοπίσει σφάλματα και να θεωρήσει τους αντισυμβαλλόμενους υπεύθυνους για τις ενέργειες τους. Κανένας συμμετέχων δεν μπορεί να αντικαταστήσει ή να πειράξει τα δεδομένα λόγο της αρχιτεκτονικής του blockchain.

(Ενότητα 4.1.β Digital Assets)

Στο blockchain υπάρχουν διάφοροι τύποι ψηφιακών περιουσιακών στοιχείων (digital assets) που αντιπροσωπεύονται, συνήθως με το μορφή ενός token. Τα tokens είναι είτε ανταλλάξιμα (το ένα μπορεί να αντικατασταθεί με το άλλο) είτε όχι (το καθένα είναι μοναδικό). Αυτά τα tokens μπορεί να είναι είτε κρυπτονομίσματα είτε αναπαραστάσεις υφισταμένων χρηματοπιστωτικών μέσων όπως τα ομόλογα. Επίσης τα tokens μπορούν να χρησιμοποιηθούν για τη διασφάλιση της αυθεντικότητας και την παρακολούθηση της ιδιοκτησίας ψηφιακών έργων τέχνης αλλά και άλλων μορφών. Ακόμη τα tokens μπορούν να αντιπροσωπεύσουν φυσικά περιουσιακά στοιχεία καθώς και τεκμηρίωσης κρίσιμης σημασίας για τις επιχειρήσεις, όπως τιμολόγια.

(Ενότητα 4.1.γ Blockchain & E-commerce)

Το blockchain μπορεί να έχει ποικίλες εφαρμογές στο ηλεκτρονικό εμπόριο, με στόχο την βελτίωσή του. Κάποια παραδείγματα είναι τα εξής:

1. Στα παραδοσιακά συστήματα επεξεργασίας πληρωμών τα οποία περιλαμβάνουν περίπου 16 βήματα και οι συνολικές χρεώσεις κυμαίνονται από 2% έως 6%. Λαμβάνοντας υπόψη τα πολυάριθμα που εμπλέκονται σε μια συναλλαγή, η απλοποίηση της διαδικασίας έχει ως αποτέλεσμα γρηγορότερες συναλλαγές, κάτι το οποίο μπορεί να ωφελήσει τόσο τους εμπόρους όσο και τους πελάτες. Καθώς μια συναλλαγή πραγματοποιείτε στο blockchain, η ανάγκη για μεσολαβητές εξαλείφεται και η ταχύτητα μιας συναλλαγής καθορίζεται από την ταχύτητα του δικτύου, καθώς και την ταχύτητα που παράγονται τα νέα blocks.

2. Μεσάζοντες όπως η επεξεργασία πληρωμής και τα λογιστικά μπορούν να ρυθμιστούν μέσω smart contracts. Επίσης η χαρτογράφηση και η οπτικοποίηση στις αλυσίδες εφοδιασμού μπορούν να βελτιωθούν σημαντικά, καθώς η δομή των δεδομένων του blockchain είναι στην ουσία μια σειρά από καταγεγραμμένα και χωρίς δυνατότητα επεξεργασίας δεδομένα. Ακόμα λόγο της φύσης του blockchain τα καταστήματα μπορούν να έχουν πλήρη ιδιοκτησία των περιουσιακών τους στοιχείων, όπως προϊόντα, φωτογραφίες, περιγραφές, βίντεο, κριτικές, ψηφιακές βιτρίνες κ.λπ. Ακόμη, οι καταναλωτές μπορούν να έχουν πρόσβαση σε πλήρεις πληροφορίες όπως η προέλευση, η επεξεργασία και τα συστατικά των προϊόντων.

3. Ένα σύστημα ηλεκτρονικού εμπορίου που χρησιμοποιεί το blockchain, δεν έχει πια την ανάγκη για πολλαπλές συνδέσεις πύλης καθώς ένας λογαριασμός είναι αρκετός για να χρησιμοποιήσει όλες τις εφαρμογές ο χρήστης.

(Υποκεφάλαιο 4.2 Σχεδιασμός ηλεκτρονικού καταστήματος)

(Ενότητα 4.2.α Χαρακτηριστικά συστήματος)

Το ηλεκτρονικό κατάστημα θα αποτελείται κυρίως από 2 συστήματα, τον blockchain server (Server , n.d.) και τον server που θα φιλοξενεί το website (Website, n.d.).

Το blockchain θα είναι υπεύθυνο για την κύρια επιχειρησιακή λογική του συστήματος, καθώς θα περιέχει τον βασικό κώδικα του συστήματος αλλά και τη δομή των δεδομένων που θα αποθηκεύονται μόνιμα στο δίκτυο του blockchain.

Το website θα είναι υπεύθυνο για την εμφάνιση της ιστοσελίδας, την σωστή πλοήγηση του χρήστη και την διατήρηση της επικοινωνίας με το blockchain, με την σωστή αλληλουχία.

(Ενότητα 4.2.β Κύρια χαρακτηριστικά εφαρμογής)

Το ηλεκτρονικό κατάστημα θα περιέχει έξι βασικά χαρακτηριστικά τα οποία είναι: η αγορά προϊόντων, η λίστα με τα διαθέσιμα προϊόντα, η επιλογή να πουλήσει οποιοσδήποτε χρήστης, η κρυφή δημοπρασία για το ποιος έμπορος θα μεταφέρει το προϊόν, η αυτοματοποίηση υπολογισμού του φόρου και ο μηχανισμός επιβεβαίωσης ότι κάθε βήμα, στην ροή, έχει πραγματικά επιτευχθεί .

Κάθε επισκέπτης στην σελίδα, θα έχει στην διάθεση του μια λίστα με τα διαθέσιμα προϊόντα και θα έχει την επιλογή να τα αγοράσει, χωρίς να υπάρχει κάποιος περιορισμός.

Οποιοσδήποτε έμπορος ή επισκέπτης, θα έχει την δυνατότητα γρήγορα και εύκολα να πουλήσει τα προϊόντα του στη σελίδα με βάση την κοστολόγηση και τα διαθέσιμα κομμάτια που θα ορίσει ο ίδιος.

Κάθε φορά που επιλέγει κάποιος χρήστης να αγοράσει ένα προϊόν από τη σελίδα χρειάζεται να διαχειριστούμε το θέμα της αποστολής του δέματος. Με στόχο να αναλάβει, το website, την διαχείριση της αποστολής θα δημιουργεί κάθε φορά μια κρυφή δημοπρασία. Έτσι οποιοσδήποτε έμπορος θέλει να αναλάβει την μεταφορά θα χρειάζεται να συμμετάσχει στην δημοπρασία. Όποιος νικήσει την δημοπρασία, θα αναλαμβάνει την μεταφορά. Στην περίπτωση που ο ίδιος πωλητής θέλει να αναλάβει την μεταφορά τότε θα χρειάζεται και αυτός να συμμετάσχει στην δημοπρασία και μόνο στην περίπτωση που νικήσει θα μπορέσει να την αναλάβει, χωρίς δυνατότητα αποφυγής αυτής της ροής. Λόγο αυτής της σχεδιαστικής απόφασης, τα μεταφορικά θα διακυμαίνονται συνήθως σε προσιτές τιμές, και θα είναι αρκετά δύσκολο να καταφέρει κάποιος έμπορος ή χρήστης την μονοπώληση των μεταφορών με αποτέλεσμα την αύξηση των τιμών.

Σε κάθε συναλλαγή που θα συμβαίνει στον χώρο της ιστοσελίδας, θα υπάρχει αφαίρεση και υπολογισμός φόρου σε μια μορφή αυτοματισμού, εν γνώσει του χρήστη, αλλά χωρίς

δυνατότητα αποφυγής. Η αυτοματοποίηση υπολογισμού του φόρου έχει ως σκοπό, την ομαλή διασύνδεση και λειτουργία της ιστοσελίδας με τον κρατικό οργανισμό.

Για να μπορούμε να επιβεβαιώσουμε ότι κάθε βήμα στην ροή της συναλλαγής έχει πραγματικά επιτευχθεί (δηλαδή να είμαστε σίγουροι ότι ο μεταφορέας όντως παρέδωσε το δέμα ή ότι ο αγοραστής όντως παρέλαβε το δέμα) θα χρειαστεί κάθε φορά που πραγματοποιείτε μια φυσική πράξη να καταγράφεται στο blockchain. Ακόμη θα χρειαστεί να παρέχουμε κάποιου είδους κίνητρο στο να καταγράφονται οι πράξεις, για να μην είναι εύκολο να «ξεγελάσει» κάποιος το σύστημα.

(Ενότητα 4.2.γ Συναλλαγές και αλληλεπιδράσεις χρηστών)

Όλες οι πιθανές ροές του χρήστη με το ηλεκτρονικό κατάστημα είναι τρείς. Ο χρήστης θα μπορεί να αγοράσει ένα προϊόν, να το πουλήσει και να δεσμεύσει κάποιο ποσό στην δημοπρασία για την μεταφορά του δέματος. Σε μια ολοκληρωμένη ροή αγοράς ενός προϊόντος υπάρχουν δύο συναλλαγές, μία όταν ο αγοραστής αγοράζει το προϊόν και μια δεύτερη όταν ο πωλητής πληρώνει τη μεταφορική.

Στο ηλεκτρονικό κατάστημα υπάρχει μόνο ένα είδος χρήστη και κανένας διαχειριστής. Ο ρόλος του χρήστη ξεχωρίζει από το μέρος του στην ροή μιας συναλλαγής, δηλαδή μπορεί να είναι είτε αγοραστής, είτε πωλητής, είτε μεταφορέας.

(Ενότητα 4.2.δ Γενικές Υποθέσεις)

Αυτό το έργο βασίζεται στις αρχές του blockchain που αναγράφονται στο white paper του blockchain (Satoshi, 2008), και η ιδέα είναι να χρησιμοποιηθούν αυτές οι αρχές στον σχεδιασμό του ηλεκτρονικού καταστήματος με σκοπό τη βελτίωσή του σε διάφορους τομείς.

(Ενότητα 4.2.ε Γενικοί περιορισμοί)

Παρόλο που η επικοινωνία του blockchain γίνεται μέσω του website, δεν μπορεί να είναι περιοριστικό ως μοναδικό μέσω. Απο την στιγμή που θα είναι ο κώδικας διαθέσιμος πάνω στο blockchain θα μπορεί να «καλεστεί» από οποιονδήποτε, με επαρκές γνώσεις, χωρίς να είναι ανάγκη να χρησιμοποιήσει το website. Σχετικά περιοριστικό σε αυτή την περίπτωση είναι το γεγονός ότι ο χρήστης που επιθυμεί να καλέσει μόνος του τον κώδικα, δεν θα έχει, πολλές φορές, αρκετά ξεκάθαρη την πρόθεση του κώδικα ή ποιο θα έπρεπε να ήταν το τελικό αποτέλεσμα. Σε περίπτωση κακής χρήσης θα εμφανίζεται σχετικό μήνυμα στο σύστημα του blockchain.

Κατά επέκταση, δεν θα μπορέσουμε να περιορίσουμε ποιος χρησιμοποιεί τον κώδικα που θα υπάρχει στο δίκτυο του blockchain.

Ακόμη, το website θα πρέπει να έχει ως στόχο την εύκολη χρήση του, όσο είναι αυτό δυνατόν, από έναν απλό χρήστη χωρίς τεχνικές γνώσεις.

ΚΕΦΑΛΑΙΟ 5 Έργο

(Υποκεφάλαιο 5.1 Εργαλεία)

(Ενότητα 5.1.α Τύπος Blockchain)

Για τον τύπο του blockchain θα χρησιμοποιήσουμε το Ethereum blockchain καθώς μας προσφέρει τα αναγκαία χαρακτηριστικά στην γλώσσα προγραμματισμού που χρειαζόμαστε όπως βρόγχους, για την υλοποίηση του έργου.

(Ενότητα 5.1.β Γλώσσα προγραμματισμού)

Κατ' επέκταση, αφού θα χρησιμοποιήσουμε το Ethereum blockchain, θα χρησιμοποιήσουμε την γλώσσα solidity για την υλοποίηση της κύριας επιχειρησιακής λογικής. Είναι μια αντικειμενοστραφής γλώσσα προγραμματισμού και έχει δημιουργηθεί με στόχο την συγγραφή των smart contract. Η Solidity έχει πάρει επιρροές από την C++, Python και JavaScript. Λόγο τον σχεδιασμό της, επιτρέπει την εύκολη υλοποίηση σεναρίων όπως εκλογές, τυφλές δημοπρασίες και πορτοφόλια υπογραφών. Συγκεκριμένα έχει χρησιμοποιηθεί η έκδοση 0.7.4 (Solidity, n.d.)

Για την υλοποίηση του website θα χρησιμοποιήσουμε την γλώσσα προγραμματισμού JavaScript (συχνά συντομογραφείται JS) όπου είναι μια ευρέως γνωστή γλώσσα για τον σκοπό αυτό. Αποτελεί μια από τις βασικές τεχνολογίες του Παγκόσμιου ιστού (World Wide Web) μαζί με την HTML και την CSS. Πάνω από το 97% των ιστότοπων χρησιμοποιούν JavaScript για την συμπεριφορά ιστοσελίδων. Όλα τα μεγάλα προγράμματα περιήγησης (π.χ. Google chrome) διαθέτουν ειδική μηχανή JavaScript για την εκτέλεση του κώδικα στις συσκευές των χρηστών. (JavaScript, n.d.)

(Ενότητα 5.1.γ Frameworks)

Χρησιμοποιώντας μόνο JavaScript για την υλοποίηση του website δεν θα είναι ούτε εύκολο αλλά ούτε και γρήγορο. Για αυτό τον λόγο θα χρησιμοποιήσουμε κάποιο framework για την ομαλή και γρήγορη υλοποίηση του website. Συγκεκριμένα θα χρησιμοποιήσουμε το framework React, όπου είναι μια βιβλιοθήκη για την γλώσσα JavaScript και έχει ως στόχο την δημιουργία των όμορφων και απλών διεπαφών του χρήστη. Ακόμη είναι μια βιβλιοθήκη ανοιχτού κώδικα και διατηρείτε κυρίως από την εταιρία Meta (πρώην Facebook) και μια κοινότητα μεμονωμένων προγραμματιστών (React, n.d.).

(Ενότητα 5.1.δ Truffle Suite)

Για την ανάπτυξη της web3 εφαρμογής θα χρησιμοποιήσουμε το truffle suite το οποίο είναι μια συλλογή από τρία διαφορετικά εργαλεία που έχουν ως σκοπό την διευκόλυνση τόσο στην ανάπτυξη smart contract όσο και στην δοκιμή τους.

Συγκεκριμένα θα χρησιμοποιήσουμε το Truffle όπου είναι ένα περιβάλλον το οποίο βοηθάει στην αυτοματοποίηση, στην δοκιμή και στην ανάπτυξη smart contracts στο Ethereum Virtual Machine. Βοηθάει στην δημιουργία απλών και εύχρηστα script όπου αντιδρούν στις αλλαγές της εφαρμογής. Επίσης, μας επιτρέπει να εξοικονομήσουμε χρόνο με το να επικοινωνούμε με τα smart contracts μέσω μιας διαδραστικής κονσόλας, ακόμη περιλαμβάνει πρόσβαση σε όλες τις διαθέσιμες εντολές Truffle.

Ένα ακόμα εργαλείο, του truffle suite, που χρησιμοποιούμε είναι το Ganache. Το Ganache μπορεί να χρησιμοποιηθεί σαν ένα προσωπικό blockchain για την γρήγορη ανάπτυξη εφαρμογών, καθώς διευκολύνει όλες τις φάσεις ανάπτυξης της εφαρμογής, όπως προγραμματισμό, στήσιμο και δοκιμή προσφέροντας ένα ασφαλή και ντετερμινιστικό περιβάλλον (Truffle Suite, n.d.) .

(Ενότητα 5.1.ε Δοκιμές)

Για τις συνεχόμενες δοκιμές των βασικών λειτουργειών θα χρησιμοποιήσουμε μια βιβλιοθήκη για δοκιμές. Συγκεκριμένα θα χρησιμοποιήσουμε την Chai, όπου είναι μια βιβλιοθήκη με στόχο να διευκολύνει τεχνικές υλοποίησης όπως, προγραμματισμό με γνώμονα τη συμπεριφορά και προγραμματισμό με βάση την δοκιμαστική ανάπτυξη. Η βιβλιοθήκη αυτή είναι υλοποιημένη για τον browser ή για node server (Chai Assertion Library, n.d.).

Επίσης θα χρειαστούμε ακόμα μια βιβλιοθήκη τη Mocha, η οποία είναι ένα JavaScript test framework συγκεκριμένα πλούσιο σε χαρακτηριστικά με σκοπό τις δοκιμές του κώδικα. Επίσης και αυτή η βιβλιοθήκη τρέχει στον browser και σε node server. Οι δοκιμές με Mocha εκτελούνται σειριακά, επιτρέποντας έτσι ευέλικτες και ακριβείς αναφορές, ενώ παράλληλα αντιστοιχίζονται με μη αποτυπωμένες εξαιρέσεις στις σωστές περιπτώσεις δοκιμών (Mocha, χ.χ.).

(Ενότητα5.1.ζ Package manager)

Για να διαχειριστούμε τις εξαρτήσεις του κώδικα σε άλλες βιβλιοθήκες, θα χρειαστούμε έναν διαχειριστή πακέτων. Συγκεκριμένα θα διαλέξουμε το npm, το οποίο είναι το μεγαλύτερο registry (μητρώο) λογισμικού στον κόσμο. Οι προγραμματιστές ανοιχτού κώδικα χρησιμοποιούν το npm για να μοιράζονται και να δανείζονται πακέτα.

Επίσης πολλοί οργανισμοί το χρησιμοποιούν για να διαχειρίζονται την ιδιωτική ανάπτυξη.

Το npm αποτελείται από τρία βασικά στοιχεία:

- Την ιστοσελίδα
- Την διεπαφή της γραμμής εντολών (CLI)
- Το registry

Η ιστοσελίδα χρησιμοποιείτε από τους χρήστες για να ανακαλύπτουν πακέτα, να ρυθμίζουν προφίλ και να διαχειρίζονται άλλες πτυχές της εμπειρίας του npm.

Η διεπαφή της γραμμής εντολών εκτελείται από ένα τερματικό και είναι ο τρόπος με τον οποίο συνήθως οι περισσότεροι προγραμματιστές αλληλεπιδρούν με το npm.

Το registry είναι μια μεγάλη δημόσια βάση δεδομένων λογισμικού JavaScript και των μετα-πληροφοριών που το περιβάλλουν (nmp docs, n.d.).

(Ενότητα 5.1.η Nodejs)

Για το περιβάλλον ανάπτυξης της ιστοσελίδας θα χρησιμοποιηθεί η βιβλιοθήκη Nodejs, που βασίζεται σε ασύγχρονα συμβάντα. Το Nodejs έχει σχεδιαστεί κυρίως για τη δημιουργία επεκτάσιμων εφαρμογών δικτύου (Node.js, n.d.).

(Ενότητα 5.1.θ web3js)

Για την επικοινωνία της ιστοσελίδας με το Ethereum blockchain θα χρησιμοποιηθεί η βιβλιοθήκη web3.js. Το web3.js είναι μια συλλογή από βιβλιοθήκες οι οποίες επιτρέπουν στον χρήστη να αλληλοεπιδράσουν με ένα τοπικό ή απομακρυσμένο κόμβο του Ethereum με την βοήθεια HTTP, IPC, και WebSocket πρωτοκόλλων επικοινωνίας (web3.js, χ.χ.).

(Ενότητα 5.1.ι MetaMask)

Για την διαχείριση των λογαριασμών και των κλειδιών στο blockchain θα χρησιμοποιηθεί το εργαλείο MetaMask. Το MetaMask επιτρέπει εύκολες αλληλεπιδράσεις των χρηστών με το web3. Δημιουργήθηκε για να καλύψει τις ανάγκες ασφαλών και χρησιμοποιήσιμων ιστοσελίδων που βασίζονται στο Ethereum. Συγκεκριμένα χειρίζεται τη διαχείριση των λογαριασμών και τη σύνδεση του χρήστη με το blockchain.

Το MetaMask προσφέρει μια μεγάλη βελτίωση ασφαλείας σε σχέση με αυτή της αποθήκευσης των κλειδιών χρήστη σε έναν μόνο κεντρικό διακομιστή ή ακόμα και σε τοπικό χώρο αποθήκευσης. Επομένως το εργαλείο αυτό μπορεί να αποτρέψει μαζικές κλοπές λογαριασμών.

Επίσης, έρχεται προφορτωμένο με γρήγορες συνδέσεις με το blockchain Ethereum και πολλά δίκτυα δοκιμής. Αυτό μας επιτρέπει να ξεκινήσουμε χωρίς να συγχρονίζουμε έναν πλήρη κόμβο, ενώ εξακολουθεί να παρέχει την επιλογή να αναβαθμίσουμε την ασφάλεια και να χρησιμοποιήσουμε τον πάροχο blockchain της επιλογής μας. Το MetaMask είναι συμβατό με οποιοδήποτε blockchain το οποίο εκθέτει Ethereum-συμβατά JSON RPC API (JSON RPC API, n.d.), συμπεριλαμβανομένων προσαρμοσμένων και ιδιωτικών blockchain. Κατά επέκταση το MetaMask είναι το ιδανικό εργαλείο για την ανάπτυξη και εκτέλεσης ενός δοκιμαστικού blockchain όπως το Ganache (MetaMask Docs, n.d.).

(Υποκεφάλαιο 5.2 Κώδικας)

(Ενότητα 5.2.α Δομές δεδομένων)

Στο smart contract υπάρχουν αρκετές μεταβλητές που χρειάζονται για την απαραίτητη λειτουργία του καταστήματος. Από τον απεικονιζόμενο κώδικα όπως φαίνεται στη εικόνα 4 και 5, θα αναλύσουμε τις πιο σημαντικές δομές.

Έχουμε μια μεταβλητή όπου θα κρατάμε τον αριθμό όλων τον προϊόντων “uint public productCount”, η οποία θα χρησιμοποιηθεί και σαν αναγνωριστικό. Θα κρατάμε την διεύθυνση του πωλητή για να έχουμε την δυνατότητα να του μεταφέρουμε τα λεφτά στο τέλος της συναλλαγής. Θα έχουμε μια δομή mapping (Mapping Types, n.d.), η οποία είναι μια δομή για να μπορούμε να έχουμε key-value τύπους και θα αποθηκεύει τα λεφτά του νικητή της κρυφής δημοπρασίας για την μεταφορά. Ακόμη μια δομή mapping που θα χρησιμοποιήσουμε είναι αυτή που θα κρατάει όλα τα διαθέσιμα προϊόντα, και ένα mapping που θα αποθηκεύονται όλοι οι εισπραττόμενοι φόροι.

Μια ακόμη βασική δομή που χρησιμοποιούμε είναι η struct δομή (Structs, n.d.) όπου θα μπορούμε να αποθηκεύουμε διάφορες μεταβλητές. Στην περίπτωση μας, έχουμε μια μεταβλητή struct “ProductWithBids” που κρατάει όλα τα βασικά δεδομένα που χρειαζόμαστε για ένα προϊόν. Επίσης θα χρησιμοποιήσουμε struct δομή για να αναπαραστήσουμε και να αποθηκεύουμε τις προσφορές που γίνονται σε κάθε δημοπρασία.

```
contract eShop {
    string public name;
    uint public productCount = 0;
    //BlindAuction
    address payable public beneficiary;//seller
    //Money for paying the transporter(winner of the auction),
    //can not be more than the item's price.
    mapping(uint => uint) public transportDeposit;
    //main mapping for products
    mapping(uint => ProductWithBids) public internalProducts;
    uint biddingTime = 120;
    uint revealTime = 60;
    /*
     * this struct serves only for storing extra
     * variables since solidity structs can not handle
     * more than 16 entries in the struct otherwise we
     * will get error:
     * Stack too deep when compiling inline assembly.
     */
    mapping(uint => AdditionalVars) private extraProductVars;
    uint public globalBidsCount = 0;
    //Taxes
    mapping(address => uint) public taxes;
    uint public taxCounter = 0;

    modifier maxValue(uint id) {
        require(
            msg.value <= transportDeposit[id],
            "Bid can not be more than item's price"
        );
        _;
    }

    struct ProductWithBids {
        uint id;
        string name;
        uint price;
        address payable owner;
        address payable beneficiary;
        bool purchased;

        mapping(address => Bid[]) bids;
        mapping(address => SecretBids[]) nakedBids;
        // Allowed withdrawals of previous bids
        mapping(address => uint) pendingReturns;

        mapping(address => uint) bidsCount;
        bool ended;
        uint biddingEnd;
        uint revealEnd;
        address payable lowestBidder;
        uint lowestBid;
    }

    struct AdditionalVars {
        uint id;
        bool firstTime;
    }

    struct SecretBids {
        uint values;
        bool fake;
        string secret;
    }

```

Εικόνα 4 Μεταβλητές του smart contract

```
73
74      struct Bid {
75          bytes32 blindedBid;
76          uint deposit;
77      }
78
```

Εικόνα 5 Μεταβλητή struct Bid του smart contract

(Ενότητα 5.2.β Συναλλαγή)

Μια ροή συναλλαγής θα ήταν ως εξής: ένας πωλητής καταχωρεί το προϊόν του στην ιστοσελίδα → ένας πιθανός αγοραστής αποφασίζει να το αγοράσει καταβάλλοντας τα χρήματα → έναρξη της κρυφής δημοπρασίας για το ποιος θα μεταφέρει το προϊόν, και αφού τελειώσει η δημοπρασία, → ο μεταφορέας παραδίδει το προϊόν και → ο πωλητής και ο μεταφορέας παίρνουν τα λεφτά που τους αναλογούν.

Ωστόσο, υπάρχει το πρόβλημα της εμπιστοσύνης μεταξύ αυτών των τριών μελών. Για παράδειγμα, αν ο παραλήπτης παραλάβει το προϊόν αλλά δεν το επιβεβαίωση, ο πωλητής και ο μεταφορέας δεν θα μπορούν να εισπράξουν τα λεφτά τους. Για αυτό, θα χρησιμοποιήσουμε έναν μηχανισμό επιβεβαίωσης σε κάθε βήμα στην ροή και ο πωλητής θα πληρώσει σαν υποθήκη την διπλάσια τιμή του προϊόντος.

Έτσι η ολοκληρωμένη ροή μιας συναλλαγής θα ήταν ως εξής:
Ο πωλητής καταχωρεί το προϊόν και υποθηκεύει την διπλάσια τιμή του προϊόντος. Αξίζει να σημειωθεί ότι ο πωλητής είναι υπεύθυνος να συμπεριλάβει την τιμή των μεταφορικών στο προϊόν, καθώς ο ίδιος πληρώνει τους μεταφορείς στο τέλος. Έπειτα, ο αγοραστής επιλέγει να αγοράσει το προϊόν, όπου τα λεφτά πλην τον φόρο μεταφέρονται στον πωλητή και ο φόρος στην ειδική μεταβλητή όπου αποθηκεύονται οι φόροι και στην συνέχεια ξεκινάει η κρυφή δημοπρασία για την μεταφορά του προϊόντος.

Η δημοπρασία έχει τρία κομμάτια, στο πρώτο ο καθένας ποντάρει το ποσό που θέλει μέχρι να λήξει ο χρόνος. Η κάθε διεύθυνση έχει την ευχέρεια να ποντάρει όσες φορές θέλει. Αφού λήξει ο χρόνος περνάμε στο δεύτερο κομμάτι και όλα τα ποσά επιστρέφονται εκτός από το μικρότερο που έχει ποντάρει ο καθένας. Στο τρίτο και τελευταίο κομμάτι νικάει το μικρότερο ποσό που έγινε μεταξύ όλων και τα υπόλοιπα επιστέφονται στους ιδιοκτήτες.

Αφού τελειώσει η δημοπρασία, ο πωλητής πληρώνει τον μεταφορέα και ο μεταφορέας μεταφέρει το προϊόν στον αγοραστή. Τέλος επιστρέφονται τα υπόλοιπα λεφτά στον πωλητή που είχε καταθέσει στην αρχή.

(Ενότητα 5.2.γ Καταχώρηση Προϊόν)

Στην καταχώρηση ενός προϊόντος, τεχνικά υπάρχουν κάποιες προαπαιτήσεις από το smart contract, συγκεκριμένα όπως φαίνεται και στην εικόνα 6, το προϊόν πρέπει να έχει αναγκαστικά κάποιο όνομα, η τιμή του προϊόντος να είναι μη μηδενική και η αξία της κατάθεσης του πωλητή να έχει τουλάχιστον διπλάσια τιμή από του προϊόν.

```
117     //noinspection UnprotectedFunction
118     function createProduct(
119         string memory _name,
120         uint _price
121     )
122     public
123     payable
124     {
125         //Require a name
126         require(bytes(_name).length > 0, "Item must have a name");
127         //Require a valid price
128         require(_price > 0, "Price must be valid");
129         //Require x2 msg.value of item's price
130         require(msg.value >= _price, "The collateral msg.value must at least be two times grater than the item's");
131         //increase product productCount
132         productCount ++;
133         /*
134          * Note how in all the functions, a struct type is assigned to
135          * a local variable with data location storage. This does not copy
136          * the struct but only stores a reference so that assignments
137          * to members of the local variable actually write to the state.
138          *
139          * Also We cannot use "internalProducts[productCount] = ProductWithBids(id, name, price , etc)"
140          * because the RHS creates a memory-struct "ProductWithBids" that contains a mapping.
141          */
142         //create the product
143         ProductWithBids storage newProduct = internalProducts[productCount];
144         AdditionalVars storage extraVars = extraProductVars[productCount];
145
146         extraVars.id = productCount;
147         newProduct.id = productCount;
148         newProduct.name = _name;
149         newProduct.price = _price;
150         newProduct.owner = msg.sender;
151         newProduct.beneficiary = msg.sender;
152         newProduct.purchased = false;
153         transportDeposit[productCount] = msg.value;
154         //trigger an event
155         emit ProductCreated(productCount, _name, _price, msg.sender, false);
156     }
```

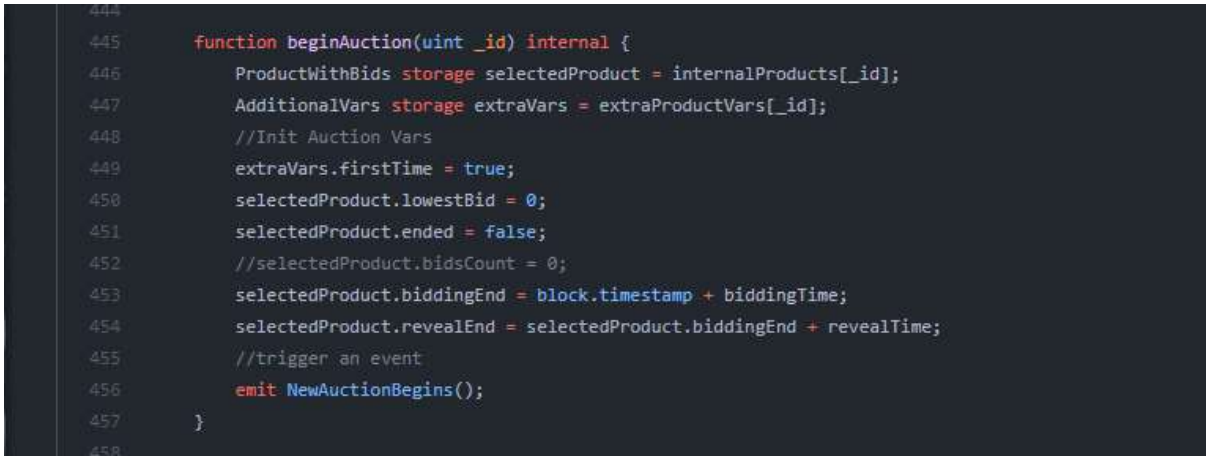
Εικόνα 6 καταχώρηση προϊόντος

(Ενότητα 5.2.δ Αγορά Προϊόν)

Στην αγορά ενός προϊόντος υπάρχουν κάποιες προαπαιτήσεις όπως, το αναγνωριστικό προϊόντος να είναι έγκυρο, η κατάθεση του αγοραστή να είναι ίδιας ή μεγαλύτερης αξίας με αυτή του προϊόντος, το προϊόν να μην έχει ήδη αγοραστεί και τέλος ο πωλητής δεν μπορεί να είναι και ο αγοραστής. Επίσης υπολογίζεται και ο φόρος του προϊόντος στην αγορά και έτσι η διαφορά στέλνεται στον πωλητή όπως φαίνεται και στην εικόνα 7. Το τέλος της αγοράς σηματοδοτεί την έναρξη της κρυφής δημοπρασίας για τον μεταφορέα όπως φαίνεται στην εικόνα 8.

```
158     function purchaseProduct(uint _id) public payable {
159         //Fetch the Product
160         ProductWithBids storage _product = internalProducts[_id];
161         //Fetch the owner
162         address payable _seller = _product.owner;
163         //Make sure the product has valid id
164         require(_product.id > 0 && _product.id <= productCount);
165         //Require that there is enough Ether in the transaction
166         require(msg.value >= _product.price, "Not enough Ether");
167         //Require that the product has not been purchased already
168         require(!_product.purchased, "The product is already purchased");
169         //Require that the buyer is not the _seller
170         require(_seller != msg.sender, "The buyer can not be the seller");
171         //Transfer ownership to the buyer
172         _product.owner = msg.sender;
173         //Mark as purchasedProduct
174         _product.purchased = true;
175         //Charge tax
176         uint tax = chargeTax(msg.value, msg.sender);
177         //Pay the seller by sending them Ether
178         _seller.transfer(msg.value - tax);
179         //trigger an event
180         emit ProductPurchased(productCount, _product.name, _product.price, msg.sender, true);
181         //beginAuction
182         beginAuction(_id);
183     }
```

Εικόνα 7 Αγορά Προϊόν



```
444
445     function beginAuction(uint _id) internal {
446         ProductWithBids storage selectedProduct = internalProducts[_id];
447         AdditionalVars storage extraVars = extraProductVars[_id];
448         //Init Auction Vars
449         extraVars.firstTime = true;
450         selectedProduct.lowestBid = 0;
451         selectedProduct.ended = false;
452         //selectedProduct.bidsCount = 0;
453         selectedProduct.biddingEnd = block.timestamp + biddingTime;
454         selectedProduct.revealEnd = selectedProduct.biddingEnd + revealTime;
455         //trigger an event
456         emit NewAuctionBegins();
457     }
458
```

Εικόνα 8 Αρχή της δημοπρασίας

(Ενότητα 5.2.ε Κρυφή Δημοπρασία)

Όταν ξεκινάει μια δημοπρασία ξεκινάει να μετράει ο χρόνος για την πρώτη φάση που κατά την διάρκεια της, και μόνο τότε, μπορεί κάποιος να ποντάρει.

```
185       /// Start of BlindAuction
186       /// In a real world scenario this function should
187       /// call only the bid function because the nakedBids
188       /// should have been an external input from another
189       /// contract or application.
190       function checkBidding(
191           uint _id,
192           bytes32 _blindedBid,
193           uint _values,
194           bool _fake,
195           string memory _secret
196       )
197       public
198       payable
199       maxValue(_id)
200       {
201           ProductWithBids storage selectedProduct = internalProducts[_id];
202           //onlyBefore biddingEnd
203           require(block.timestamp < selectedProduct.biddingEnd, "The bidding has closed");
204
205           bid(_blindedBid, _id);
206           pushNakedBids(
207               _values,
208               _fake,
209               _secret,
210               _id
211           );
212           selectedProduct.bidsCount[msg.sender]++;
213       }
214
215       /// Place a blinded bid with `_blindedBid` =
216       /// keccak256(abi.encodePacked(value, fake, secret)).
217       /// The sent ether is only refunded if the bid is correctly
218       /// revealed in the revealing phase. The bid is valid if the
219       /// ether sent together with the bid is at least "value" and
220       /// "fake" is not true. Setting "fake" to true and sending
221       /// not the exact amount are ways to hide the real bid but
222       /// still make the required deposit. The same address can
223       /// place multiple bids.
224       function bid(
225           bytes32 _blindedBid,
226           uint id
227       )
228       public
229       payable
230       {
231           ProductWithBids storage selectedProduct = internalProducts[id];
232           //onlyBefore biddingEnd
233           //require(block.timestamp < newProduct.biddingEnd);
234
235           selectedProduct.bids[msg.sender].push(Bid({
236           blindedBid : _blindedBid,
237           deposit : msg.value
238           }));
239
240           globalBidsCount++;
241       }
```

Εικόνα 9 Ποντάρισμα στην δημοπρασία

Όταν τελειώσει η πρώτη φάση και προχωρήσουμε στην δεύτερη φάση, όλα τα πονταρίσματα επιστρέφονται εκτός από το μικρότερο που έχει καταβάλει ο καθένας, όπως φαίνεται στην εικόνα 10.

```
    /// The main function responsible for revealing
    /// the bids. In a real world scenario this
    /// function should be a separate contract or
    /// app, because this contract simulates an
    /// external input.
    function reveal(uint _id) public payable {
        ProductWithBids storage selectedProduct = internalProducts[_id];
        //onlyAfter biddingEnd && onlyBefore revealEnd
        require(block.timestamp > selectedProduct.biddingEnd, "Only after the bidding has closed");
        require(block.timestamp < selectedProduct.revealEnd, "The reveal phase has ended");
        uint count = selectedProduct.bidsCount[msg.sender];

        uint[] memory values = new uint[](count);
        bool[] memory fake = new bool[](count);
        string[] memory secret = new string[](count);

        for (uint i = 0; i < count; i++) {
            values[i] = selectedProduct.nakedBids[msg.sender][i].values;
            fake[i] = selectedProduct.nakedBids[msg.sender][i].fake;
            secret[i] = selectedProduct.nakedBids[msg.sender][i].secret;
        }

        revealInternal(
            values,
            fake,
            secret,
            _id
        );
    }
```

Εικόνα 10 Αποκάλυψη πονταρισμάτων

```
275     /// Secret is just the required string for the matching encoding.
276     /// Reveal your blinded bids. You will get a refund for all
277     /// correctly blinded invalid bids and for all bids except for
278     /// the totally highest.
279     function revealInternal(
280         uint[] memory _values,
281         bool[] memory _fake,
282         string[] memory _secret,
283         uint _id
284     )
285     public
286     payable
287     {
288         ProductWithBids storage selectedProduct = internalProducts[_id];
289         //onlyAfter biddingEnd && onlyBefore revealEnd
290         require(block.timestamp > selectedProduct.biddingEnd, "Only after the bidding has closed");
291         require(block.timestamp < selectedProduct.revealEnd, "The reveal phase has ended");
292
293         uint length = selectedProduct.bids[msg.sender].length;
294         require(_values.length == length, "invalid length value");
295         require(_fake.length == length, "invalid length fake");
296         require(_secret.length == length, "invalid length secret");
297
298         uint refund;
299         for (uint i = 0; i < length; i++) {
300             Bid storage bidToCheck = selectedProduct.bids[msg.sender][i];
301             (uint value, bool fake, string memory secret) =
302             (_values[i], _fake[i], _secret[i]);
303             if (bidToCheck.blindedBid != keccak256(abi.encodePacked(value, fake, secret))) {
304                 // Bid was not actually revealed.
305                 // Do not refund deposit.
306                 emit NotRevealedYet(msg.sender);
307                 continue;
308             }
309             refund += bidToCheck.deposit;
310             if (fake) {
311                 emit isTrue(fake);
312             } else {
313                 emit isFalse(fake);
314                 emit deposit(bidToCheck.deposit);
315                 emit depositValue(value);
316             }
317             if (!fake && bidToCheck.deposit >= value) {
318                 if (placeBid(msg.sender, value, _id))
319                     refund -= value;
320             }
321             // Make it impossible for the sender to re-claim
322             // the same deposit.
323             bidToCheck.blindedBid = bytes32(0);
324         }
325         msg.sender.transfer(refund);
326         //emit event
327         emit Revealed(msg.sender);
328     }
```

Εικόνα 11 Εσωτερική μέθοδος αποκάλυψης πονταρισμάτων

```
329
330     /// Withdraw a bid that was overbid.
331     function withdraw(
332         uint _id
333     )
334     public
335     payable
336     {
337         ProductWithBids storage selectedProduct = internalProducts[_id];
338         uint amount = selectedProduct.pendingReturns[msg.sender];
339         if (amount > 0) {
340             // It is important to set this to zero because the recipient
341             // can call this function again as part of the receiving call
342             // before `transfer` returns .
343             selectedProduct.pendingReturns[msg.sender] = 0;
344
345             msg.sender.transfer(amount);
346         }
347     }
348
```

Εικόνα 12 Απόσυρση προσφορών που ξεπεράστηκαν από κάποιες άλλες

```
459     /// This is an "internal" function which means that it
460     /// can only be called from the contract itself (or from
461     /// derived contracts).
462     function placeBid(
463         address payable bidder,
464         uint value,
465         uint id
466     )
467     internal
468     returns (bool success)
469     {
470         ProductWithBids storage selectedProduct = internalProducts[id];
471         AdditionalVars storage extraVars = extraProductVars[id];
472
473         //TODO this is reversed.
474         if (value >= selectedProduct.lowestBid && !extraVars.firstTime)
475             return false;
476         else
477             extraVars.firstTime = false;
478
479         if (selectedProduct.lowestBidder != address(0)) {
480             // Refund the previously highest bidder.
481             selectedProduct.pendingReturns[selectedProduct.lowestBidder] += selectedProduct.lowestBid;
482         }
483         selectedProduct.lowestBid = value;
484         selectedProduct.lowestBidder = bidder;
485         emit IWillReturnTrue();
486         return true;
487     }
```

Εικόνα 13 Εσωτερική μέθοδος προσφορών

Έπειτα φτάνουμε στη τρίτη και τελευταία φάση της δημοπρασίας όπου αποκαλύπτεται ο νικητής, όπως φαίνεται στην εικόνα 14. Όσες προσφορές χάσανε στην δημοπρασία, μπορούν να εισπραχθούν από τους ιδιοκτήτες τους πατώντας το κουμπί “withdraw”. Η μέθοδος της υλοποίησης αυτής φαίνεται στην εικόνα 12. Στην συνέχεια ο πωλητής εισπράττει αυτόματα την αμοιβή του, πλην τους φόρους όπως φαίνεται στην εικόνα 15.

```
348
349     /// End the auction and send the highest bid
350     /// to the beneficiary.
351     function auctionEnd(uint _id) public payable {
352         ProductWithBids storage selectedProduct = internalProducts[_id];
353         //onlyAfter revealEnd
354         require(block.timestamp > selectedProduct.revealEnd, "Only after the reveal phase has ended");
355
356         require(!selectedProduct.ended, "The auction has ended already");
357         emit AuctionEnded(
358             selectedProduct.lowestBidder,
359             selectedProduct.lowestBid
360         );
361
362         // It is important to place this if because the recipient
363         // can call this function again as part of the receiving call
364         // before `transfer` returns .
365         if (transportDeposit[_id] - selectedProduct.lowestBid >= 0 && !selectedProduct.ended) {
366             selectedProduct.lowestBidder.transfer(selectedProduct.lowestBid + selectedProduct.lowestBid);
367             uint tax = chargeTax(selectedProduct.lowestBid, selectedProduct.beneficiary);
368             uint sub = selectedProduct.lowestBid + tax;
369             transportDeposit[_id] -= sub;
370         }
371
372         refundSeller(
373             selectedProduct.beneficiary,
374             _id
375         );
376         selectedProduct.ended = true;
377     }
378
```

Εικόνα 14 Τέλος δημοπρασίας

```
409     function refundSeller(
410         address payable _seller,
411         uint _id
412     )
413     public
414     payable
415     {
416         uint amount = transportDeposit[_id];
417         if (amount > 0) {
418             if (amount > 0)
419                 _seller.transfer(amount);
420             // It is important to set this to zero because the recipient
421             // can call this function again as part of the receiving call
422             // before `transfer` returns .
423             transportDeposit[_id] = 0;
424         } else if (amount < 0) {
425             //emit event about the state
426             //of transportDeposit[_id].
427         }
428     }
429
430     function chargeTax(
431         uint amount,
432         address payable buyer
433     )
434     public
435     payable
436     returns (uint tax)
437     {
438         tax = amount * 24 / 100;
439         if (taxes[buyer] == 0)
440             taxCounter++;
441         taxes[buyer] += tax;
442         return tax;
443     }
```

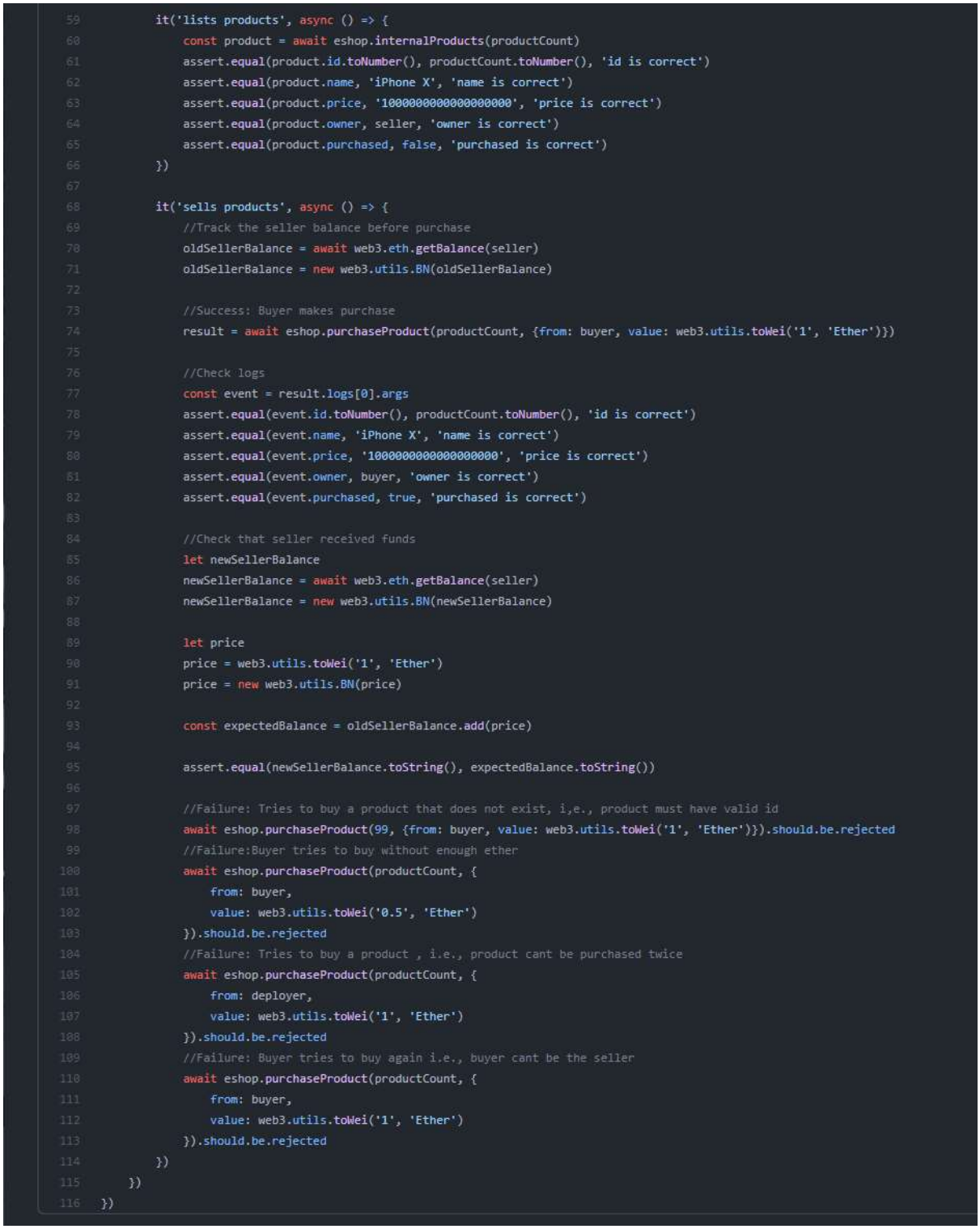
Εικόνα 15 Αυτόματη πληρωμή του πωλητή του προϊόν

(Ενότητα 5.2.ζ Κώδικας δοκιμών)

Κάθε smart contract χρειάζεται να περνάει από κάποιες αυτόματες δοκιμές, όπως φαίνεται στις εικόνες 16 και 17, για να σιγουρέψουμε όσο γίνεται ότι λειτουργεί ο κώδικας με τις προθέσεις που φτιάχτηκε.

```
const eShop = artifacts.require('./eShop.sol')

require('chai')
    .use(require('chai-as-promised'))
    .should()

/*
 * Basic testing using the chai framework,
 * to ensure the contract has been deployed
 * successfully.
 */
contract('eShop', ([deployer, seller, buyer]) => {
    let eshop

    before(async () => {
        eshop = await eShop.deployed()
    })

    describe('deployment', async () => {
        it('deploys successfully', async () => {
            const address = await eshop.address
            assert.notEqual(address, 0x0)
            assert.notEqual(address, '')
            assert.notEqual(address, null)
            assert.notEqual(address, undefined)
        })

        it('has a name', async () => {
            const name = await eshop.name()
            assert.equal(name, 'Dapp eShop')
        })
    })

    describe('products', async () => {
        let result, productCount, bidTime

        before(async () => {
            result = await eshop.createProduct('iPhone X', web3.utils.toWei('1', 'Ether'), {from: seller})

            productCount = await eshop.productCount()
        })

        it('creates products', async () => {
            //Success
            assert.equal(productCount, 1)
            const event = result.logs[0].args
            assert.equal(event.id.toNumber(), productCount.toNumber(), 'id is correct')
            assert.equal(event.name, 'iPhone X', 'name is correct')
            assert.equal(event.price, '1000000000000000000', 'price is correct')
            assert.equal(event.owner, seller, 'owner is correct')
            assert.equal(event.purchased, false, 'purchased is correct')

            //Failure : Product must have a name
            await eshop.createProduct('', web3.utils.toWei('1', 'Ether'), {from: seller}).should.be.rejected;
            //Failure : Product must have a price
            await eshop.createProduct('iPhone X', web3.utils.toWei('0', 'Ether'), {from: seller}).should.be.rejected;
        })
```

Εικόνα 16 Κώδικας δοκιμών λειτουργιών smart contract



```
59          it('lists products', async () => {
60              const product = await eshop.internalProducts(productCount)
61              assert.equal(product.id.toNumber(), productCount.toNumber(), 'id is correct')
62              assert.equal(product.name, 'iPhone X', 'name is correct')
63              assert.equal(product.price, '100000000000000000', 'price is correct')
64              assert.equal(product.owner, seller, 'owner is correct')
65              assert.equal(product.purchased, false, 'purchased is correct')
66          })
67
68          it('sells products', async () => {
69              //Track the seller balance before purchase
70              oldSellerBalance = await web3.eth.getBalance(seller)
71              oldSellerBalance = new web3.utils.BN(oldSellerBalance)
72
73              //Success: Buyer makes purchase
74              result = await eshop.purchaseProduct(productCount, {from: buyer, value: web3.utils.toWei('1', 'Ether')})
75
76              //Check logs
77              const event = result.logs[0].args
78              assert.equal(event.id.toNumber(), productCount.toNumber(), 'id is correct')
79              assert.equal(event.name, 'iPhone X', 'name is correct')
80              assert.equal(event.price, '100000000000000000', 'price is correct')
81              assert.equal(event.owner, buyer, 'owner is correct')
82              assert.equal(event.purchased, true, 'purchased is correct')
83
84              //Check that seller received funds
85              let newSellerBalance
86              newSellerBalance = await web3.eth.getBalance(seller)
87              newSellerBalance = new web3.utils.BN(newSellerBalance)
88
89              let price
90              price = web3.utils.toWei('1', 'Ether')
91              price = new web3.utils.BN(price)
92
93              const expectedBalance = oldSellerBalance.add(price)
94
95              assert.equal(newSellerBalance.toString(), expectedBalance.toString())
96
97              //Failure: Tries to buy a product that does not exist, i,e., product must have valid id
98              await eshop.purchaseProduct(99, {from: buyer, value: web3.utils.toWei('1', 'Ether')}).should.be.rejected
99              //Failure:Buyer tries to buy without enough ether
100             await eshop.purchaseProduct(productCount, {
101                 from: buyer,
102                 value: web3.utils.toWei('0.5', 'Ether')
103             }).should.be.rejected
104             //Failure: Tries to buy a product , i.e., product cant be purchased twice
105             await eshop.purchaseProduct(productCount, {
106                 from: deployer,
107                 value: web3.utils.toWei('1', 'Ether')
108             }).should.be.rejected
109             //Failure: Buyer tries to buy again i.e., buyer cant be the seller
110             await eshop.purchaseProduct(productCount, {
111                 from: buyer,
112                 value: web3.utils.toWei('1', 'Ether')
113             }).should.be.rejected
114         })
115     })
116 })
```

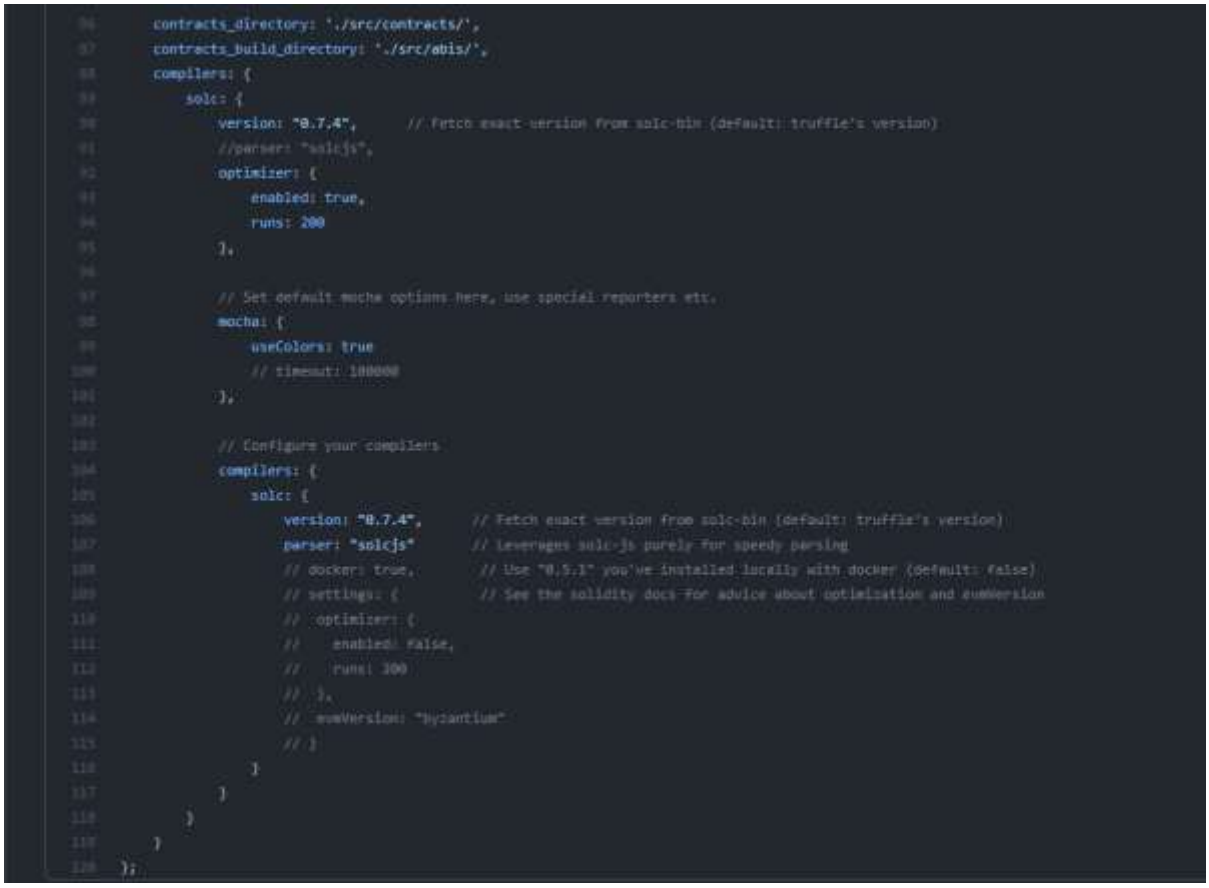
Εικόνα 17 Κώδικας δοκιμών λειτουργιών smart contract

(Ενότητα 5.2.η Ρυθμίσεις Truffle)

Για να μπορέσουμε να χρησιμοποιήσουμε το framework truffle θα χρειαστεί να θέσουμε κάποιες απαραίτητες ρυθμίσεις όπως φαίνεται στην εικόνα 18 και 19. Αξίζει να παρατηρηθεί στην εικόνα 19, ότι έχουμε διευκρινίσει σε ποιο σημείο στον κώδικα βρίσκονται τα smart contracts και σε πιο σημείο θα παραχθεί το αποτέλεσμα του κώδικα μετά τη μετάφραση (Compiler, n.d.) σε κώδικα μηχανής συμβατή με το EVM. Επίσης διευκρινίζουμε ποια εκδοχή της γλώσσας Solidity θα χρησιμοποιηθεί.

```
module.exports = {
    /**
     * Networks define how you connect to your ethereum client and let you set the
     * defaults web3 uses to send transactions. If you don't specify one truffle
     * will spin up a development blockchain for you on port 9545 when you
     * run `develop` or `test`. You can ask a truffle command to use a specific
     * network from the command line, e.g
     *
     * $ truffle test --network <network-name>
     */

    networks: {
        // Useful for testing. The `development` name is special - truffle uses it by default
        // if it's defined here and no other network is specified at the command line.
        // You should run a client (like ganache-cli, geth or parity) in a separate terminal
        // tab if you use this network and you must also set the `host`, `port` and `network_id`
        // options below to some value.
        //
        development: {
            host: "127.0.0.1",     // Localhost (default: none)
            port: 8545,            // Standard Ethereum port (default: none)
            network_id: "*",       // Any network (default: none)match any network id
            // url: "https://BlockChainEshop.com",
            https: true,
            homepage: "https://BlockChainEshop.com",
            dns: "https://BlockChainEshop.com"
        },
```

Εικόνα 18 truffle ρυθμίσεις για το δίκτυο



Εικόνα 19 truffle ρυθμίσεις, θέτοντας το σημείο που βρίσκονται τα smart contracts και ποια εκδοχή της γλώσσας Solidity χρησιμοποιούμε

(Ενότητα 5.2.θ Ρυθμίσεις npm)

Για να μπορούμε να διαχειριστούμε τις εξαρτήσεις του κώδικα σε άλλες βιβλιοθήκες χρησιμοποιούμε τον package manager npm. Το npm χρησιμοποιεί ένα αρχείο "package.json" για να διαβάσει τις ρυθμίσεις που χρειάζεται όπως φαίνεται στην εικόνα 20 και 21.

```
  "name": "eshop-blockchain",
  "version": "2.0.0",
  "private": "true",
  "description": "An ethereum marketplace",
  "homepage": "http://BlockChainEshop.com",
  "main": "truffle-config.js",
  "dependencies": {
    "babel-polyfill": "6.26.0",
    "babel-preset-env": "1.7.0",
    "babel-preset-es2015": "6.24.1",
    "babel-preset-stage-2": "6.24.1",
    "babel-preset-stage-3": "6.24.1",
    "babel-register": "6.26.0",
    "bn.js": "^4.11.9",
    "bootstrap": "^3.4.1",
    "browserslist": "^4.16.3",
    "chai": "^4.2.0",
    "chai-as-promised": "7.1.1",
    "chai-bignumber": "3.0.0",
    "ethereumjs-util": "^7.0.8",
    "ethers": "^5.0.31",
    "js-sha3": "^0.8.0",
    "react": "16.8.4",
    "react-bootstrap": "^1.5.2",
    "react-dom": "16.8.4",
    "react-scripts": "^4.0.3",
    "rxjs": "^6.6.3",
    "sha3": "^2.1.3",
    "solc": "^0.7.6",
    "truffle": "^5.1.52",
    "typescript": "^4.2.3",
    "web3": "^1.3.4",
    "web3-utils": "^1.3.4"
  },
  "scripts": {
    "start": "react-scripts start",
    "start-deploy": "open-cli http://UthEshop:3001 && node bin/www",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject",
    "stop": "pkill --signal SIGNIT eshopBlockchain"
  },
  "eslintConfig": {
    "extends": "react-app"
  },
  "directories": {
    "test": "test"
  },
  "browserlust": [
    ">0.2%",
    "not dead",
    "not ie <= 11",
    "not op_mini all"
  ],
  "keywords": [],
  "author": "GeorgePapadopoulos",
  "license": "ISC",
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
```

Εικόνα 20 Αρχείο package.json ρυθμίσεις

```
64        ],
65        "development": [
66          "last 1 chrome version",
67          "last 1 firefox version",
68          "last 1 safari version"
69        ]
70      },
```

Εικόνα 21 Αρχείο package.json ρυθμίσεις

ΚΕΦΑΛΑΙΟ 6 Συμπεράσματα

Η τεχνολογία blockchain έχει την δυνατότητα να φέρει μεγάλες αλλαγές στον τρόπου που φτιάχνονται σήμερα η web εφαρμογές. Η συγκεκριμένη πτυχιακή εργασία δημιουργήθηκε με σκοπό να αναδείξει τα οφέλη αλλά και τα αρνητικά της τεχνολογίας αυτής. Όπως δείξαμε στο κεφάλαιο 5, η ανάπτυξη των smart contracts απαιτεί προσοχή σε κάθε βήμα καθώς, σε περιπτώσεις λάθους μπορεί να καταλήξει να είναι πολύ δαπανηρό. Έτσι μια αποκεντρωμένη προσέγγιση στην ανάπτυξη λογισμικού μπορεί να αποφέρει πολλά οφέλη, μεταξύ των οποίων είναι η αύξηση της αποτελεσματικότητας με την κατάργηση του μεσάζοντα και την προσθήκη του αυτοματισμού, μεγαλύτερη διαφάνεια στα δημόσια δίκτυα, μεγαλύτερη ασφάλεια με τη χρήση κρυπτογραφίας και βελτιωμένη ιχνηλασιμότητα με αμετάβλητα αρχεία. Παρόλα αυτά είναι σημαντικό να κατανοήσουμε ότι το blockchain ενώ είναι πολλά υποσχόμενο, δεν μπορεί να λύσει κάθε πρόβλημα. Για αυτό είναι αρκετά σημαντικό να μπορούμε να διευκρινίσουμε σε ποιες περιπτώσεις δεν είναι συνετό να χρησιμοποιήσουμε blockchain. Για παράδειγμα αν χρειαζόμαστε αμετάβλητα δεδομένα για την περίπτωση χρήσης μας, καθώς τα δεδομένα του blockchain δεν μπορούν να αλλάξουν. Αν δουλεύουμε με ψηφιακά στοιχεία, διότι αν όχι, είναι πολύ πιθανόν να μην χρειαζόμαστε blockchain. Βέβαια, υπάρχουν και αμφιλεγόμενες περιπτώσεις. Αυτές αφορούν περιπτώσεις κατά τις οποίες δεν είναι σίγουρο το αν πρέπει να χρησιμοποιηθεί το blockchain ή όχι. Αυτό συμβαίνει σε περιπτώσεις που απαιτούνται συναλλαγές με πολύ υψηλές αποδόσεις. Παρόλο που το blockchain δεν μπορεί να αποδώσει τόσο όσο τα παραδοσιακά προγράμματα, δεν μπορεί να είναι ο καταλύτης της απόφασής μας. Αυτό συμβαίνει γιατί τις περισσότερες φορές, αν και όχι τόσο αποδοτικά όσο τα παραδοσιακά προγράμματα, το blockchain μπορεί να καλύψει τις περισσότερες ανάγκες για μεγάλο όγκο συναλλαγών. Για την συγκεκριμένη εφαρμογή υπάρχουν πολλά περιθώρια βελτίωσης, όπως μια πιο «φιλική» σελίδα διεπαφής σε μη τεχνικούς χρήστες ή ακόμα και στην γραφή του κώδικα που μπορεί να γραφτεί καλύτερα και να αξιοποιήσει πλήρως τα χαρακτηριστικά της γλώσσας.

ΒΙΒΛΙΟΓΡΑΦΙΑ

{Xie, W. a. (2018). *ETTF: A Trusted Trading Framework Using Blockchain in E-commerce.* IEEE .

Ahmed S. Almasoud, F. K. (2020). Smart contracts for blockchain-based reputation systems: A systematic literature review. *Journal of Network and Computer Applications*.

Arjun Rachana Harish, X. L. (2021). Log-flock: A blockchain-enabled platform for digital asset valuation and risk assessment in E-commerce logistics financing. *Computers & Industrial Engineering*.

Buterin, V. (2013). Ethereum Whitepaper. *GitHub repository*, 22-23.

Chai Assertion Library. (χ.χ.). Ανάκτηση από Chai Assertion Library: https://www.chaijs.com/

Compiler. (χ.χ.). Ανάκτηση από wikipedia: https://en.wikipedia.org/wiki/Compiler

Enterprise resource planning. (χ.χ.). Ανάκτηση από wikipedia: https://en.wikipedia.org/wiki/Enterprise_resource_planning

Genesis block. (χ.χ.). Ανάκτηση από wikipedia: https://en.bitcoin.it/wiki/Genesis_block

Gulshan Kumar, R. S.-H. (2020,). Decentralized accessibility of e-commerce products through blockchain technology. *Sustainable Cities and Society*.
Hemantkumar P. Bulsara, P. S. (2020). Blockchain Technology for E-commerce Industry. *International Journal of Advanced Science and Technology*.
Hongmei, Z. (2021). A Cross-Border E-Commerce Approach Based on Blockchain Technology. *Mobile Information Systems*.
JavaScript. (χ.χ.). Ανάκτηση από wikipedia: https://en.wikipedia.org/wiki/JavaScript
Jim, M. (1998). *Structured Query Language*. Ανάκτηση από wikipedia: https://en.wikipedia.org/wiki/SQL
JSON RPC API. (χ.χ.). Ανάκτηση από Ethereum wiki: https://eth.wiki/json-rpc/API
Lahkani MJ, W. S. (2020). Sustainable B2B E-Commerce and Blockchain-Based Supply Chain Finance. *Sustainability*.
Mapping Types. (χ.χ.). Ανάκτηση από soliditylang: https://docs.soliditylang.org/en/v0.8.12/types.html#mapping-types
MetaMask Docs. (χ.χ.). Ανάκτηση από metamask.io: https://docs.metamask.io/guide/#why-metamask
Ming Li, S. S. (2020). Blockchain-enabled logistics finance execution platform for capital-constrained E-commerce retail. *Robotics and Computer-Integrated Manufacturing*.
Mocha. (χ.χ.). Ανάκτηση από Mocha: https://mochajs.org/
nmp docs. (χ.χ.). Ανάκτηση από Npm: https://docs.npmjs.com/about-npm
Node.js. (χ.χ.). Ανάκτηση από Node.js: https://nodejs.org/en/
React. (χ.χ.). Ανάκτηση από reactjs: https://reactjs.org/
Roberto Tonelli, A. P. (2018). Ethereum smart contracts as blockchain-oriented microservices. *Association for Computing Machinery*.
S. Wang, L. O.-Y. (2019). Blockchain-Enabled Smart Contracts: Architecture, Applications, and Future Trends. *IEEE Transactions on Systems*.
Satoshi, N. (2008). Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*.
Server . (χ.χ.). Ανάκτηση από wikipedia: https://en.wikipedia.org/wiki/Server_(computing)
Shuiguang Deng, G. C. (2020). Incentive-Driven Computation Offloading in Blockchain-Enabled E-Commerce. *ACM Trans*.
Solidity. (χ.χ.). Ανάκτηση από soliditylang: https://docs.soliditylang.org/en/v0.8.12/index.html
Structs. (χ.χ.). Ανάκτηση από soliditylang: https://docs.soliditylang.org/en/v0.8.12/types.html#structs
Sun, Y. a. (2020). RTChain: A Reputation System with Transaction and Consensus Incentives for E-Commerce Blockchain. *Association for Computing Machinery*.
Syed, T. A. (2019). A comparative analysis of blockchain architecture and its applications: Problems and recommendations. *IEEE* .
Truffle Suite. (χ.χ.). Ανάκτηση από Truffle Suite: https://trufflesuite.com/index.html
web3.js. (χ.χ.). Ανάκτηση από Ethereum JavaScript API: https://web3js.readthedocs.io/en/v1.7.0/
Website. (χ.χ.). Ανάκτηση από wikipedia: https://en.wikipedia.org/wiki/Website
what are smart contracts. (χ.χ.). Ανάκτηση από ethereum: https://ethereum.org/en/developers/docs/intro-to-ethereum/#what-are-smart-contracts
what is ether. (χ.χ.). Ανάκτηση από ethereum.org/: https://ethereum.org/en/developers/docs/intro-to-ethereum/#what-is-ether
Zhili Zhou, M. W.-N. (2021). Blockchain-based decentralized reputation system in E-commerce environment. *Future Generation Computer Systems*.

Zhiyong Liu, Z. L. (2020,). A blockchain-based framework of cross-border e-commerce supply chain. *International Journal of Information Management*.