Project ONYX

Software Engineering Set Free

Official Whitepaper

Zain Admani

October 17, 2017

Contents

1 Introduction

1.1 Overview

ONYX presents the first decentralized software development marketplace that is powered by the Ethereum blockchain. Software engineers are expensive to employ and are not generally used to their full potential. Every software engineering project has architectural work that is difficult and heavily relies on the talents of employed engineers to be completed correctly. However, projects also have many menial tasks that need to be implemented in order to accomplish the high level goals. These menial tasks, while necessary, are in many cases simple and a waste of time for an expensive software engineer who could be working on more difficult tasks. ONYX allows you to put these menial jobs on a marketplace where other software engineers can do them for money. ONYX also provides the ability for those who are not good coders or new to a specific language to be able to architect a solution and leave the implementation up to the ONYX network.

ONYX provides the unique opportunity for large corporations, small startups, and even individuals to create complex software by creating the high level architecture and then offloading chunks of development work to a crowd of software engineers. These engineers can simply be paid for the work that they do instead of hiring new full-time employees. Using this tactic, we hope that software can be created cheaper and faster than current practices.

1.2 Distributed Test Driven Development

ONYX is able to leverage the blockchain to crowd source software development by relying on the verifiability of test driven development. In test driven development, functions are first defined by writing unit tests that fully specify the inputs and outputs for a function without actually implementing it. Once a function is fully specified, the implementation can be flexible as long as it satisfies the tests.

In this paradigm, the tests provide a simple way to conclude if the development of a function is done properly. ONYX is able to leverage this fact in order to offload the implementation of a function to a crowd of software engineers safely. This permits your core engineers to focus on defining more features and high level characteristics. The code written by the crowd of engineers is guaranteed to be correct in accordance to the tests. This allows for expensive software engineers to do the more difficult tasks of designing large systems while still getting the implementation done in parallel leading to savings in time and money.

The many companies that already apply test driven development principles in their software engineering could get an instant benefit from ONYX without significantly effecting their workflow.

1.3 Long Term Vision

A mature version of ONYX would resemble the JIRA boards that are currently so widely used in the industry. A corporation/startup will design a project by breaking it into manageable chunks that are added to the board and adding test cases. Each chunk of work on the board (with its test cases) will be sent to ONYX and implemented automatically. Work is completed in parallel as it is added to the board leading to efficient development. It will also have hooks

that will allow a team to attach it to any workflow management tool that the team currently uses. A mature version of ONYX should be able to match work supply and demand perfectly without getting in the way.

The long term vision of ONYX is to let a handful of engineers to be able to do the work that would currently take dozens of engineers to complete. This will enable talented engineers to pursue projects that they otherwise wouldn't because of starting costs and time. In time this may lead to completely decentralized software development companies where a few engineers design the product and then offload the implementation work to ONYX.

2 ONYX System Architecture

2.1 Major Players

The ONYX system is architected around 3 main actors: Requesters, Engineers, Validators.

2.1.1 Requesters

Requesters are the main drivers of the ONYX economy and send the requests for software engineering work to be completed. A request consists of the function/object header and an array of tests that must be run on the final work to be sure that it meets the specification. In addition to this, the Requesters attach the number of ether they are willing to pay an Engineer to complete the task.

Our vision is that Requesters will be composed of a mix of software development corporations, startups, and open source community. These entities all have the most to gains in terms of work needed to be completed at the lowest price and fluctuating work demands.

2.1.2 Engineers

Engineers are the users in the ONYX network that look for tasks to complete in exchange for ether. Engineers attempt to take the requests from Requesters and pass all the tests given. Once an Engineer feels that they have successfully passed all of the tests, they submit the finished code for validation by a random Validator in the network. If the Validator returns a failing result then this process is repeated.

Engineers can pick and choose the work they do in addition to where and when they do it. Since Engineers get to choose the requests they pick, they can also pick similar requests and reuse code across different requests in order to make the process even more efficient. This has benefits for both Engineers and Requesters since Engineers get more payments for less work and Requesters get faster completion times.

2.1.3 Validators

Validators are the approvers that allow Requesters to know without a doubt that the request has been completed properly (all tests passing) and unlock the payments to the Engineers. Validators get paid a small fee from the ether payment to the Engineers for this service. All code is immutable and unreadable by the Validators for security purposes. The goal for the Validator process flow is to be highly automated to the point where a user simply has to run an executable on their machine that waits on the ONYX network and automatically catches and

validates code submissions from Engineers without any human intervention. This is necessary since it would also promise a consistent testing environment for all the code in the network and for all parties. Validators allows anybody to participate in ONYX and make money even if they are not technically inclined. This would open up ONYX to a broader audience and make it much more valuable.

2.2 Architecture Diagram

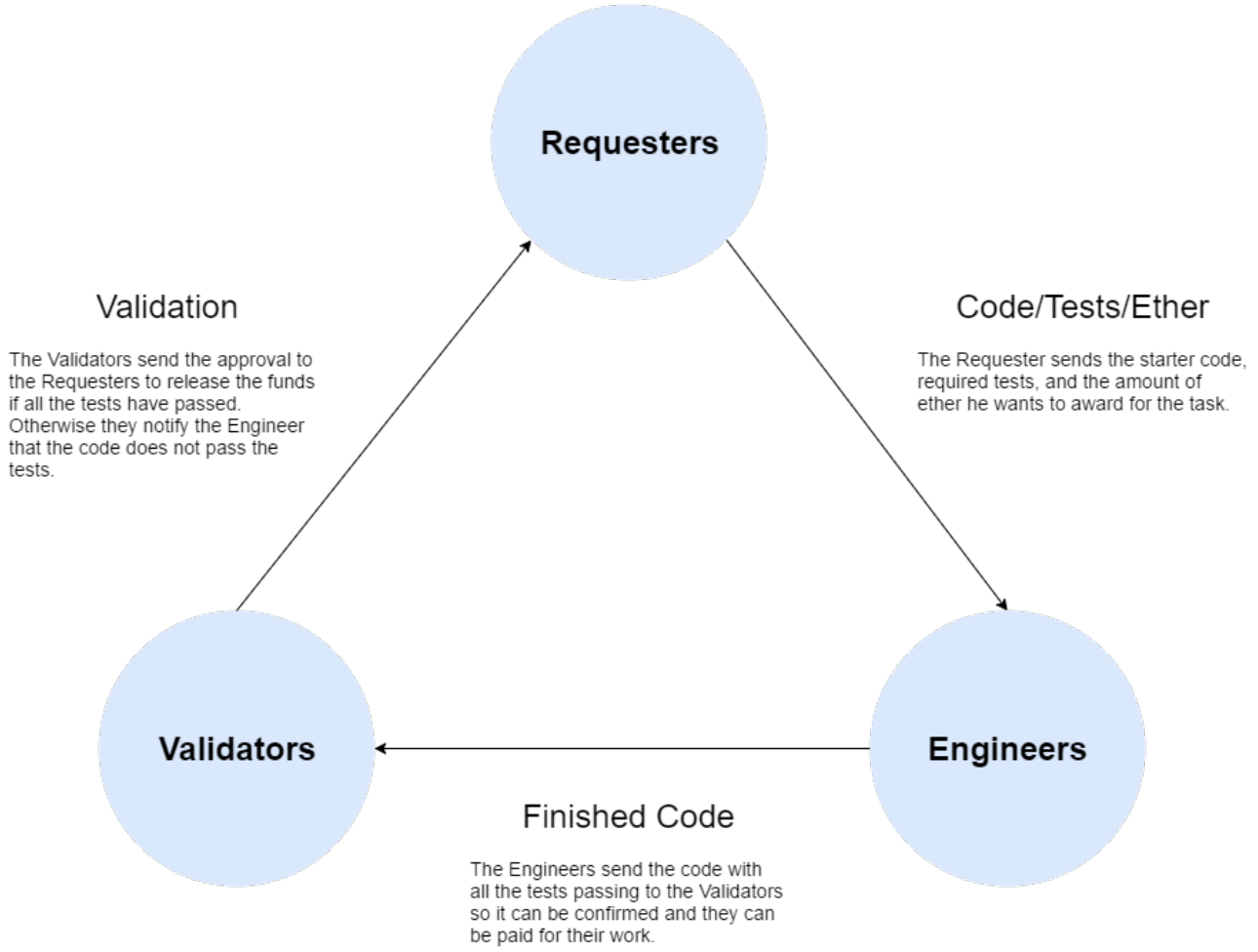


Figure 1: Visual Diagram of flow of code and ONYX tokens

2.3 Storage

All the files for the code and tests will be stored in a decentralized fashion using IPFS. This will allow for the Ethereum blockchain to simply hold the hash to the actual location of the files on IPFS. Since most ONYX jobs are expected to be on the scale of a few days in length, the duration of storage should not be too long in order to keep costs down. These small necessary storage fees will be paid by the users when conducting a transaction.

2.4 Fee

There will be a small fee that is attached to transactions on the ONYX network that goes to the developer team and corporation. This is to create a revenue stream to finance updates

and future editions. This fee will be set by the developers and will be periodically updated to respond to any future changes and market conditions.

2.5 ONYX Token

The ONYX token will be fundamental to the future operation and success of the ONYX network. After the crowdsale, tokens will be further circulated on a small market built on the website. The token will hold the following functions that will ensure that the network produces high quality code and nobody abuses their position.

2.5.1 Stake

ONYX tokens will be required in order to use the network. Each actor participating in the ONYX network is required to stake a minimum number of ONYX tokens for each transaction. The ONYX tokens will be held in a smart contract until the transaction (request to response) is completed. This minimum number of ONYX tokens will initially be set by the developers but after a short period of time (less than 1 year after the crowdsale) will be put to vote for all the token holders. A vote will take place if 20% of token holders call for it within a 2 week time window and will be the average of the responses by the token holders.

Historically, voting governance mechanisms have been plagued with low voter turnout causing a few people to decide changes for the whole network. To overcome this, we introduce the idea of the status quo default. All non-voters will be defaulted as voting for the current effective value for the stake. This will ensure 2 things:

1. Not voting will naturally lead to nothing changing which seems to be the most rational outcome.

2. If a small percentage of users vote in an extreme way, the majority of status quo votes will buffer any applied change from being too extreme.

Stake is meant to accomplish a few things. Since only token holders may use ONYX, it makes it so that every actor in the network has motivation to see ONYX succeed. A stakeholder wouldn't want to harm the network since they would be losing money if the network falters. Stake also makes it so that harmful actors cannot harm ONYX without repercussion using jury committees (read below for more details).

2.5.2 Distributed Jury

The major overall concern with ONYX is the fact that code written by an external actor is being run on personal machines. An Engineer could slip in some malware to be executed along with accomplishing the task. So in this situation, the tests would pass but the code would definitely not be desirable. A similar case could be made where the Requester slips in malware in the test file when giving it to the Engineer. This problem will be mitigated by the ONYX token stake.

In this situation, the Requester will call for a complaint against the Engineer within a short time frame (24 hours) upon receiving the code (Ether will still be in escrow and ONYX tokens will still be at stake). After a complaint is filed, it is placed onto the ONYX network where token holders can select to become jurors and review the complaint. If a complaint is deemed

to be true, the guilty party loses their stake and it is split between all the jurors and the complaining party. The ether is returned back to the Requester. If a complaint is deemed to be false, the false accuser loses their stake and it is split between all the jurors and the falsely accused. The ether is sent to the Engineer.

2.5.3 Voting

ONYX token holders will receive notable voting powers within the ONYX network for governance. These governance abilities are meant to help the network evolve over time to match changing needs of the users and also deal with malicious actors. These powers will include the following.

- Voting to change the stake limit which will be fundamental in controlling the balance between the number of users and the quality of users.
- Jury Voting to handle disputes and limit abuse on the ONYX network.

2.5.4 App Token

The ONYX token is seen as an app token that is used as entrance into the ONYX ecosystem. Each individual participating must stake a certain amount of ONYX tokens so it is viewed as simply a transferable membership account. Token holders do not receive dividends from fees collected on the network as might be the case in a security token. The ONYX token is only intended to be created to have utility on the ONYX network and not have intrinsic value or part ownership of the network.

2.6 Code Quality

In crowd-sourcing code implementation, there is an immediate question of code quality and coding standards being followed. We attempt to address those concerns by including a static code analyzer step in Validation in order to produce a score on the quality of code. Requesters can then also specify the level of quality that is necessary for the code they ask for and compensate accordingly. We would also like Requesters to be able to make their own custom code style constraints in order to satisfy any personal stylistic standards.

Another concern is that nobody in the Requesting party has read the code being written so it will have to be reviewed which takes time. While this concern is valid, even code that is written internal to a company is reviewed as well which mitigates the impact of this concern. The impact on time spent in code reviews is not expected to be large especially considering the code standards checks. We are still also exploring other motivation options such as reputation scores that will allow good Engineers to get more business and make them easy to identify. Reputation schemes have worked in the past in the domain of online sales such as ebay and they may be able to be effective here as well.

2.7 Safety

In the model proposed above there are clearly some safety risks that need to be mitigated.

2.7.1 Safety from Validators

One major concern is of Validators who simply "approve" all code written by any Engineer to collect their payments from the Requesters without doing their job. To mitigate this concern, a passphrase system is created.

When creating a new request, a Requester also generates a random passphrase that will be outputted when all their tests have passed. This passphrase can either be stored by the Requester locally or hashed and stored on the blockchain itself. This passphrase is then stored in a file which is not readable by either Engineer or Validator.

When an Engineer is working/testing their code on their own system, a mock file is given so that the real passphrase is not revealed but the Engineer will still know if they passed all the tests. When the Engineer passes the code to a Validator, the real read-locked passphrase file will be used. If all the tests pass, the Validator will see the results and be given the passphrase to be sent back to the Requester.

Upon receiving the results from the Validator, the Requester can easily verify that the Validator in fact did validate the results by checking if the passphrase is the same as they stored. If a matching passphrase is received, it means that the Validator definitely tested the code and the code passed all of the tests, otherwise an incorrect code would be received.

The opposite case in which Validators always "reject" all code written is not a major concern since there is a direct financial incentive to "accept" code in order to receive a payment. If a certain Validator chooses to "reject" all code, then the next randomly chosen Validator has a chance to "approve" and take the payment which would mitigate this behavior.

2.7.2 Safety from Engineers

Another major concern is of Engineers altering test files so that any code they write passes the tests when passing it along to the Validators. However, this is simply mitigated by always using a fresh copy of the tests (original copy from the Requester) when validating by the Validators. Since all code/tests are immutable by the Validators, the same concern does not exist for them.

2.7.3 Safety from Requesters

The only real concern about Requesters is the sending of malicious code to an Engineer that might harm the recipients computer like a virus or a worm. However, stake and the jury system solve this concern.

There are no other major security concerns from the Requesters. The payments are handled by smart contracts so that they can never hold off on a payment if all the tests have passed. It is up to the Requester to write good and thorough test cases in order to satisfy their use cases.

3 Roadmap

Much has to be accomplished before the long term vision and promise of ONYX can be attained. Considering that a different validation flow has to be written for each language/environment and their constraints in mind, this will be done in phases. Each of these milestones will be followed by a full audit of the code completed.

3.1 Transaction Smart Contracts

The first fundamental phase of the ONYX network is the creation of the group of smart contracts that will control the transactions between Requesters, Engineers, and Validators. The lists of contracts is roughly as follows.

- Smart contract to handle transaction between the Requester and Engineer to hold ether and ONYX token stake
- Smart contract between Engineer and Validators to handle submitting code
- Smart contract between Validator and Requesters to handle code approval
- Passcode security system to keep Validators honest
- Security read-locks on necessary files in the flow to make sure entire system is secure.

These are the basis from which the entire network can be built and are the first major milestone that must be accomplished. On top of these smart contracts, different validation flows can be implemented and added in order to handle multiple languages and environments. These are the following milestones.

3.2 Python 3.5

Python will be the first language to have its validation flow built and tested. This will be the crucial proof of concept to establish the ONYX network and give real value to the token holders. Python was chose for its popularity, relative safety, and portability (compared to C/C++ and Java). This milestone includes:

- Fully automated validation flow that will run as a daemon on a Validator computer.
- Full integration with the rest of the ONYX network.
- Set of Python 3.5 libraries that will enable passcode safety for Python 3.5 code.
- Set of standards and practices that will allow for easy implementation of test cases in a consistent and readable manner.

3.3 Java 8

The milestone targets for Java 8 will be similar to the Python 3.5 targets. However, the language itself will bring its own challenges considering it will be a jump from a scripting language to a compiled (to bytecode) language. The lessons learned from the Python 3.5 milestone should make the Java 8 milestone manageable. This milestone will be substantial since a large portion of the software companies work in Java and will start seeing benefits from ONYX.

3.4 C/C++

The final major milestone is for C/C++. Considering C/C++ are low level languages that may cause issues in both security and compatibility across many different systems will make this the most challenging milestone. The major challenges that need to be tackled for this milestone are:

- Increased security risk caused by C/C++ being very low level and working with pointers. Increased risk for buffer overflow attacks.
- Maximize compatibility across different configurations of machines (Windows/Linux/OSX).

3.5 Other Languages

Plans for languages past C/C++ are not yet known. It is thought that by the time that these 3 major languages are accomplished, the tools and practices will be established in order to quickly implement new languages relatively quickly and easily. One possible route is to put all future languages to a vote for all the token holders to decide.

3.6 Alpha Build

The development team is aiming to release an alpha build of the product onto the Ethereum testnet by mid-September. Our goal is to use the alpha build as a proof of concept that the ideas we present are valid and can work as described. This alpha is designed to be the minimum viable product for ONYX. We will release our code for bug bounties and feedback as soon as we an following this release. It will also allow us to work out any problems that have not been found yet and show curious investors what type of product they will be investing in.

3.7 Some Details in Progress

As we continue to talk to companies and interested parties to get more feedback, we are continually learning more about our potential users that lead us to changing small details of how ONYX functions. These potential changes will be made public on our channels of communication namely slack. We hope to rapidly take all the information we are receiving and build out a product that is as helpful to our users as possible and change how software is written.

4 Crowdsale

The launch of the ONYX network, and the corresponding token creation process, are organized around smart contracts running on Ethereum. These contracts will control the transfer of funds between interested parties and the ONYX team. While these details are our current goal in our crowdsale campaign, market changes or new information may cause some deviation in the final details. These changes will be made public via slack as soon as they happen.

4.1 Specifications

See Table 1. Note that all details are subject to change slightly in response to how the price of ether changes over time and our own needs as we grow.

# ONYX per 1 ether	10,000 ONYX
Minimum ether	25,000 ETH
Maximum ether (soft-cap)	60,000 ETH
Maximum ether (hard-cap)	70,000 ETH
% ONYX held by the developers	30%
Start Date (Tentative)	Feb. 10, 2018
End Date (Tentative)	Mar. 10, 2018

Table 1: Specifications for ONYX Crowdsale

4.2 Crowdsale Protocol

The crowdsale will be implemented as a smart contract on the Ethereum blockchain. The contribution period will tentatively start on Feb 10th, 2018 and will continue for approximately 4 weeks or 24 hours after the soft-cap limit is broken.

- The conversion rate of an ONYX token is static at 10,000 ONYX for 1 ether and will remain constant over the entire crowdsale period.
- Any ether sent to the crowdsale contract before or after completion of the crowdsale will not be accepted.
- The number of ONYX tokens created will vary based on the amount of funds raised.
- The crowdsale will be such that regardless of how many coins are bought, 30% of the ONYX coins will be allocated to the development team. This is because ONYX tokens give substantial powers to token holders through voting and in early stages of the project, a large percentage of the tokens must be owned in order to ensure the project gets to a self-sufficient point.
- ONYX tokens will be locked up for the duration of the crowdsale plus an additional week.
- If the minimum level of funding is not reached, then the ether will be made available back to the contributors by the smart contract using a refund function.
- More information about the crowdsale will be released out on all communication channels (shown at the end of the paper) closer to the date.

4.3 Even Distribution

A known problem in crowdsales is wealthy actors purchasing large quantities of tokens and not leaving enough for smaller actors to buy. This is especially a problem for ONYX since it relies heavily on many people having tokens to be Engineers. To solve this we follow many other crowdsales which use a soft-cap model. The crowdsale with a rather high hard ceiling (for safety from collecting excessive funding) and a lower soft-cap. Once the soft-cap is breached, the crowdsale will end in 24 hours which will let any small actors who want to get in a chance to contribute before the 24 hours is up or the hard ceiling is hit.

4.4 Budget

The money raised in the crowdsale will be raised to pay for the many costs associated with creating an application on technology that is still yet to be fully proven in a fully production setting in the wild. With this in mind, the ONYX project should be seen as an R&D project and investing in ONYX contains the same risks involved. Regardless of the funding level (between min and max), the intention is to complete all of the milestones. The funding levels will heavily influence the rate at which they can be completed however since it strongly impacts how many developers we can hire in parallel.

4.4.1 Development - 40%

The largest portion of the budget will be used to further engineer the ONYX network. This includes hiring more Ethereum engineers, front-end engineers, network specialists, and security specialists. This portion will also include paying for critical code audits and keeping the code base up to date with the most recent security and design patterns. We expect to at least double or triple the amount of people on the team in order to quickly accomplish our milestones.

4.4.2 Marketing - 20%

Considering the ONYX network heavily relies on having many users at the same time in order to take off and be successful, marketing is key. This is especially true for getting many corporations and small startups into the ONYX network to show the true value of the service. Enterprise users can fuel growth for the ONYX network faster than any other users so considerable effort will be expended to get them on board and get ONYX in their sights. Another key piece is marketing towards software engineers that ONYX is an easy way to make money for their talents and fueling growth in the Engineer part of the network.

4.4.3 Expansion & Operations - 15%

ONYX was started on the founder's laptop in his apartment. While it has expanded to include a small group of people who work out of their own homes to develop the product, it is clear much more expansion will be needed in order to meet the final vision for ONYX. This will include office space, hardware, and everything else that is required for a functioning corporation to run smoothly on a daily basis.

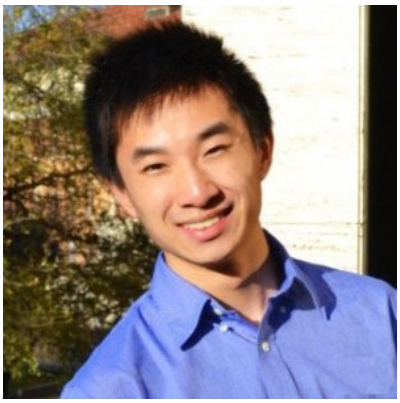
4.4.4 Legal - 25%

The team developing ONYX has always and will always maintain compliance with US law. New cutting-edge technology such as the Ethereum blockchain constantly provides legal questions that must be worked through in order to stay within the bounds of the law. Since the United States is probably the largest market for this service, we expect to be dealing with many legal hurdles on the way to making ONYX a success and we are allocating a large portion of the funding for handling these hurdles. The budget is allocated for this so that this will not become an issue and the team and the investors do everything in a legal and transparent manner.

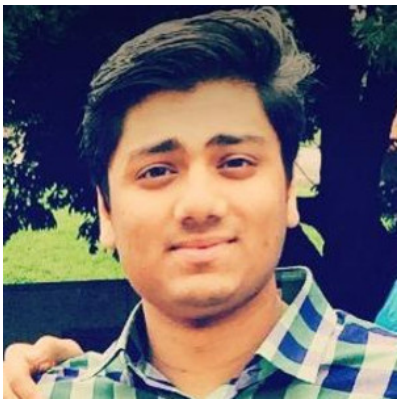
5 Development Team

5.1 Team Members

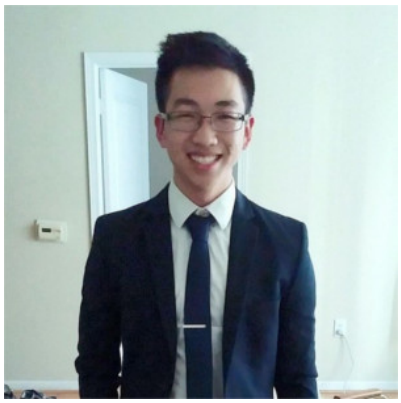
(a) Zain Admani: **Founder and Lead Developer**, M.S. Computer Science from the University of Texas focusing on AI and Deep Learning with work experience from IBM Watson and Intel



(b) Kevin Jiang: **Product Manager**, B.B.A. in M.I.S. from the University of Texas with work experience as a Product Manager at Indeed



(c) Neel Patel: **Corporate and Financial Manager**, M.S. in Accountancy at the University of Houston, C.P.A candidate with work experience at PWC



(d) Richard Huynh: **Developer**, B.B.A. in M.I.S. from the University of Texas with work experience at Trend Micro

(e) Eric Nam: **Developer**, B.B.A. in M.I.S. from the University of Texas with work experience at PWC

5.2 Open Source Development

We at Project ONYX believe that open source code is vital in being transparent and benefits the whole community. Not only that, we would love to collaborate with our community in ensuring that our code is secure and optimized. We look forward to all the feedback!

All code and development for Project ONYX is **open source** under the GPL license and can be found on Github at https://github.com/Project-ONYX/ONYX.

5.3 Communication Channels

The ONYX team strives to keep open communication channels and be transparent to all investors and people interested in the technology. We offer the following channels to communicate:

- Official Website: https://www.projectonyx.io
- Official Alpha: https://alpha.projectonyx.io
- Official Slack: https://project-onyx-slackin.herokuapp.com/
- Official Twitter: https://twitter.com/projectONYXdev
- Official Subreddit: https://www.reddit.com/r/projectonyx/
- Official Blog: https://medium.com/project-onyx