The Blockchain Based Daily Pixel Art Competition

Bachelor Thesis

Kees Fani
Manuel Borba da Silva Falcão Ferreira
Pavel Hoogland

25/06/2018

TU Coach: Dr. Ir. Rafael Bidarra
Client adviser: MSc. Marco van Etten
Coordinator: Dr. Ir. Otto Visser

TU Delft

Table of Contents

PREFACE

We would like to use this section as an opportunity to thank those that supervised and helped us during the span of the project.

First we would like to thank dr. ir. Rafael Bidarra for taking on this project and thus help make this project possible. We would also like to thank MSc. Marco van Etten for his involvement and guidance in the project.

We also want to thank YX Impact Innovations for the fun time and the nice lunches we had, and for letting us use their office space to work from.

We would also like to thank Camiel Kuiper for making some nice graphics for the game buttons, logo and infosheet.

We would also like to thank Kees' sister, Lana Fani MD for her help in shaping the thesis into a well thought out report.

We would also like to thank Romke Bak for his help on some of the mathematical calculations of the token distribution within the game.

We encountered many challenges and difficulties in the development of the software. Some of the more notable hurdles were setting up both the EOS test environment and the Progressive Web Application. The EOS environment was constantly being updated, and while setting everything up we had many errors that were not well documented. After a lot of trial and error, we were able to overcome most of these hurdles. It however did cost us time we would have preferred to have spent on actual development or software testing.

This project has been a very informative and memorable experience. We are proud of what we made and we will gladly do more projects like this in the future.

Summary

During the project the team aimed to make a decentralized web based game on the blockchain, that is fun to play and is indistinguishable from traditional centralized games. To accomplish this it was chosen to create the game on the EOS blockchain platform. The EOS blockchain platform features great improvements upon older more popular platforms such as Bitcoin and Ethereum. EOS brings improvements in transaction speed, scalability and control of user actions and data.

Designing the game was done in a somewhat limited design space. This was caused mainly due to intricacies and limitations of blockchain technology. As such, the design was steered away from games such as fast paced actions games or games that rely heavily on randomness. This is due to the transaction times of the blockchain and the reliance on determinism of the blockchain respectively.

The creation of the game consisted of two parts. One was the web based application client and the other is the EOS smart contract. The smart contract was the place where all the game logic was programmed in, while the web application is used as an interface that makes it easy for players to communicate with the blockchain smart contract.

For the development a Scrum approach was used. Well thought out software development practices were used, such as the use of Github issues and pull requests and code reviews.

The game was tested in two phases. The first phase was with an early prototype to test the pixel art drawing with competition element. The second phase was with a minimal viable product to test the overall blockchain gameplay experience and the collaborative experience.

From the gameplay testing rounds we came to the conclusion that the game, that we have created, accomplished the goal of creating a fun and engaging decentralized application that is indistinguishable from traditional centralized games.

Introduction

Cryptocurrencies have gained growing attention in the recent years by large increases in prices. Bitcoin, the most popular cryptocurrency, reached peak prices of $20.000 (D. Morris, 2017), taking along with it the whole cryptocurrency market. However, many of the people talking about cryptocurrencies only speak about price and speculation and fail to mention the technological aspects of it and the possibilities these provide.

The technology behind these cryptocurrencies is blockchain. Aside from price and speculation, blockchain is now showing potential to change the way traditional online platforms work. As of now it is still the question whether Blockchain technology will revolutionize the internet and how fast this will happen. At the turn of the millenium, something similar to the current crypto craze happened with the Dot-com bubble (E. Ofek, 2003). Now, 20 years later, many of the survivors from this Dot-com craze, such as Amazon, Booking.com, Microsoft, have become some of the biggest companies in the world and are changing the way we live our lives. This may show that people should not be so fast to underestimate the blockchain craze as well.

Blockchain Developments

Recent developments (B. Asolo, 2018) of blockchain technology have made way for more complex decentralized applications to be created. Decentralization is a way to make everyone have a share in the company and where there is no central authority with all the power on how the application will be run. Consequently, when using decentralized applications, everything is stored and processed on the blockchain.

The original decentralized blockchain implementation is Bitcoin which was created and introduced in 2009. It attempts to solve the centralization problem in payments of big banks and corporations. New blockchain platforms have since been created with improvements to many of Bitcoin's flaws such as its enormous energy consumption. We believe some of the most promising, platforms are Ethereum, EOS and Steem. Ethereum aims to improve upon the Bitcoin implementation by providing smart contracts. Whereas EOS and Steem focus on providing a blockchain platform for the creation of Decentralized Applications (DApps). Several DApps have been shown to be working, such as Steemit, DTube, IDEX and CryptoKitties. (N. Bowles, 2017)

New possibilities

Despite the rise of such new platforms, there is still much room for new possibilities in the blockchain domain. One big up and comer right now in the blockchain space is the EOS platform. EOS has come out with big promises (C. Mappster, 2017) of improvements of transaction times, governance systems with Delegated Proof of Stake (DPOS) and many more. More about this will be outlined in the Blockchain part of the report. EOS is one of the newest platforms. As of writing, its mainnet blockchain, which is the full production network, as opposed to its test network, has only been released just a week ago. Even though EOS is a new

and experimental technology, it has already garnered a lot of popularity and interest. It has in a short period of time, risen to one of the top 5 cryptocurrencies (N. Kith, 2018).

The popularity of EOS stems partly from the recognition that EOS potentially provides new opportunities for companies to develop user-friendly decentralized applications. Therefore, our client is interested in exploring the possibilities of EOS. To explore the EOS blockchain, we decided to develop a decentralized game on EOS. Games allow more freedom in its development, enabling the choice of complexity and granting the opportunity to conduct projects of smaller scale in contrast to most blockchain projects. Furthermore, gaming on the blockchain has still been barely explored. At the moment of writing, no real big games have been produced that have been built using blockchain technology, making the development of a game using this technology more challenging, exciting and promising.

Against this background, the aim of this thesis is to explore the possibilities of gaming on the blockchain.

Client and Supervision

This project has been initiated for the client *YX Impact Innovations*. YX Impact Innovations aims to develop concepts with social impact, by combining the latest innovations with creativity. The company was founded under the name of Easycollect Services in 2015, when they introduced a collection box that supports donations by contactless payments. They soon became the market leader in contactless donations in The Netherlands, as major Charity Foundations started using the innovative collection box. They are currently developing several concepts, among them a mobile application for debt prevention and an online comparison platform for retailers. YX Impact Innovations sees blockchain technology as an important innovation that may enable new and better solutions to existing problems. The project has been performed under the supervision of MSc. M. van Etten from YX impact innovations and Dr. Ir. A. R. Bidarra from the Computer Graphics and Visualisation Group.

Figure 1: Logo of YX Impact Innovations, the client of the BEP

Structure

This thesis is structured such that a person with some technical knowledge, will be able to understand every step of the process that the team went through. First in the Problem definition and analysis sections we discuss the problem we are trying to tackle. After this the Game Design phase is discussed. In this section we outline the problems we encountered during the Game Design as well as the limitations the use of blockchain would put on the design. Further is explored what the best game concepts would be to get a successful result from the project. Following this in the Software Overview section we will show an overview of the software used and how it is built up. This overview will help the reader understand the inner workings of the software and code base structure. After this we will talk about the development process of the project, here we shall also discuss the software we used to improve this process. Then in the Game Experience Testing section, we will explain the testing process for both the game experience and the software itself. After this we will give an overview of the features and structure of the game in the Delivered Prototype section. After this, in the Discussion and Recommendations section explores some of the ethical implications of our work. Here we will also what recommendations we would give the client. After this we will have a final conclusion and a reflection on the project.

Problem Definition

There have been many advancements in blockchain technology over the years (M. Crosby, 2016) with promising projects (J. Bardinelli, 2018) being announced every so often. However, the XY Impact Innovations have observed that very few of these projects have actually been successful in becoming more than just another investment. We find that most projects have only been used for speculation to quickly get high return on investments on tokens. This creates the problem of the projects themselves not being used for their intended purpose.

Game Design is one of the great challenges of the project. Therefore we determined that it is not appropriate to preemptively list the must/should/could haves (MoSCoW). These shall implicitly become apparent in the Game Design phase. The requirements are also stated in the Game Design Phase section.

Decentralized vs Centralized

The client seeks opportunities to apply the blockchain technology to have impact in a relevant field. One of the biggest possibilities blockchain provides is to have a decentralized application instead of the traditional centralized applications. Typical Centralized applications are applications such as Facebook, Google or Twitter. These big companies have all the power and are the main benefitters of success of the application. The traditional system needs a team that creates the application. They collect funding from investors and shareholders either by being part of a company or attracting outside investors. While attracting investments to create the application may sound promising, this development team will have to please the wishes of the shareholders while also keeping their user base of the application satisfied. Conflict of interest would be a potential risk here. Decentralization provides a solution for this problem by making all end-users shareholders.

Conflicts in Gaming

An interesting field for blockchain is the gaming industry. Within this industry, there is an increasing tendency towards making high profits from their players. This happens through the addition of micro-transactions to the game or by having (sometimes overpriced) Downloadable Content (DLC) added to the game. Recent examples, such as the controversy (L. Martin, 2017) around the game producer EA's Star Wars: Battlefront II DLC prices, have led to dissatisfaction among players. This is an example of a conflict of interest. The aim of the shareholders of the game is to maximize financial returns, while the player base wants an enjoyable experience where they have access to most of the content in the game.

Creation of a Blockchain Game

Since the team just consists of three bachelor Computer Science students, we cannot tackle the entirety of the decentralization / centralization problem on our own. This project aims to explore the possibilities of decentralization in gaming. The team aims to explore this by creating a simple, but fun game that is a good fit for decentralization. The research question therefore is:

Is it possible to create a decentralized game that is actually fun and feels like a 'regular' (centralized) game?

With answering this research question, the challenge to creating a game using blockchain that is understandable and fun for players becomes apparent. As a user, you should not particularly notice that the game is built on a blockchain. The game should reflect the advantages of blockchain while limiting the complications that blockchain may bring.

Problem Analysis

In this section we will analyse the problem in the following ways; first we will explain what there is to gain for consumers by switching from centralization to decentralization. We will also show what possibilities blockchain gives for gameplay elements. Further we will analyze the current state of blockchain 'games' and point out some of its current problems. Finally, we will go more in depth about our research question, by briefly explaining why we choose EOS and how we will make the game engaging. During this analysis the team will also discuss some of the challenges that are inherent to the problem.

Cost of Centralization

Centralization has a cost attached to it for consumers. This is caused by a fundamental difference between the business models of centralized and decentralized businesses. In centralized businesses, profit from the business will either be reinvested into the company or given to shareholders as dividends. In decentralized businesses, there usually is no concept of 'profit' in this manner. Value added to the business usually translates into higher priced assets in the decentralized application e.g. rise in the price of tokens. A token here refers to a cryptocurrency specific to that decentralized application.

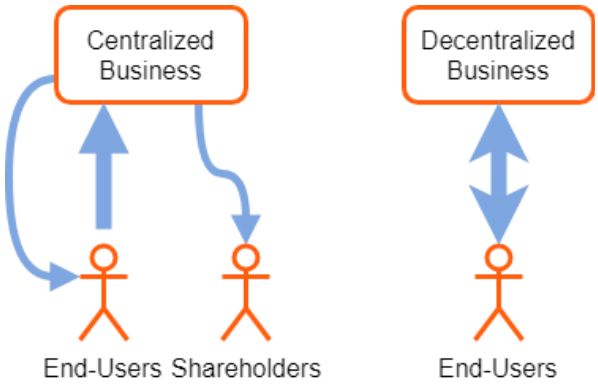


Figure 2: Overview of value flow between both centralized and decentralized systems.

Persistent Games

A problem with many traditional games is that they rely heavily on whether or not developers are willing to keep supporting them. In large scale centralized online games, when a company decides to quit production of the game, it will either have to be shut down or be overtaken by another producer or game studio. This means that the life of a centralized online game is dependent on its creators. Blockchain based games do not suffer from this problem. In the case of decentralized games, the creators/developers create a product publish it on the blockchain for everyone to use. From there on the product will become its own entity. The game is no longer in possession of the development team. The game, once it has been created, is in possession of its community. If the development team chooses to quit, the game will live on.

Other Benefits of Decentralization

Decentralization also has other benefits for us as developers, we do not need a central server that handles the game logic. Additionally the token based system encourages people to earn more by playing more such that they can trade the tokens in for real money if they like to. This can open up even newer possibilities for people to earn money while playing a game.

Challenges of the Problem

Current blockchain applications suffer from a slow and complicated user experience. Some of the features that give the blockchain its security are not particularly user friendly. An example that clearly showcases these issues is CryptoKitties (M. Taggart 2017). CryptoKitties is a digital cat collection game on the Ethereum blockchain. In it you can trade your cats and breed them with other cats to create kittens with new traits. This game had a sudden spike of popularity as people saw it as one of the first player oriented games on the blockchain. However it also exposed a lot of problems with making games on the Ethereum blockchain. CryptoKitties clogged the Ethereum Blockchain as a result of all the transactions due to its immense popularity (BBC, 2017).

On top of this, we think such ‘games’ do not really feel like ‘real games’, we think it feels more like a digital asset collection application on the blockchain. Our solutions to these blockchain game specific problems, will be addressed further in the thesis.

Blockchain

This section lays out the evolution of blockchain starting from the very beginning up until now. Here we will mostly highlight the shift the team believes in. We believe this evolution entails the shift from Bitcoin to Ethereum to EOS. We will also discuss the basic workings of the blockchain as well as some more features that have been introduced over the years. We shall also briefly discuss different *distributed consensus algorithms* as well as outlining their flaws.

Bitcoin

Some ideas for digital distributed money were being proposed in 1998 (W. Dai, 1998). However an actual real world implementation came only in 2009 when Satoshi Nakamoto (an assumed pseudonym) created Bitcoin (S. Nakamoto, 2009). Nakamoto had created a trustless digital currency. Bitcoin was launched just after the economic crisis of 2007-2008 (S. Baghla , 2017). Right at the moment that all the banks who were partly responsible for the crisis were being bailed out by governments. Through this the centralization of big banks and corporations had become clear.

Bitcoin had become the first blockchain supported currency. The blockchain is composed of blocks which are just lists of transactions. Each block in the chain contains a hash referring to the previous block. This chain of hashes goes on all the way to the first block in the chain: the Genesis block. Every node in this network needs to reach consensus on the validity and order of the blocks. The algorithm that accomplishes this is called a *consensus algorithm*. On average once every ten minutes, a new block is added to the blockchain, containing transaction that people have made on the blockchain network. A transaction is said to be `executed' when it is contained in a block.

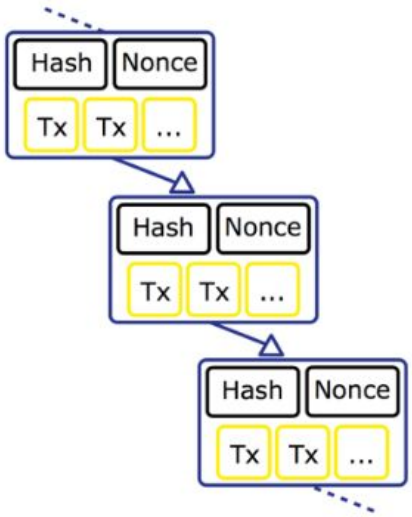


Figure 3: A number of blocks in a blockchain,
this figure also shows hashes that refer to previous blocks.

Mining

The blocks are produced when so called ‘miners’ produce pack transaction into a block in a process called ‘mining’. This means that these miners have to solve a cryptographic SHA-256 hashing function. Before a new block is broadcasted, it needs a so-called Proof of Work (PoW). Roughly speaking, this is a number, a ‘nonce’, such that *`hash(nonce + hash(block_data)) < difficulty’*. Since a (cryptographically secure) hash function is impossible to reverse, except by brute force, a miner is forced to check thousands of permutations before a correct number is found. Adding this nonce to a block proves that some amount of computing power has been performed to add the block. A block is only deemed valid when the nonce is found and attached to the block. When a miner produces a block, a set amount of bitcoin is created and awarded to the miner as a reward for solving the computation.

Ethereum and Smart Contracts

As of 2015 a new blockchain has been created that seeks to improve on Bitcoins implementation: Ethereum. Ethereum implements a PoW blockchain with a new feature: smart contracts. Smart contracts was an important new feature in blockchain technology, which was one of the reasons why Ethereum has become the second biggest cryptocurrency and blockchain platform. Essentially, a smart contract is a small program put on the blockchain. Authorised users can send transactions to this ‘contract’ to interact with it. The contract is forever preserved in the blockchain and visible to everyone. This means that the response of sending a transaction to the contract is always known, meaning that there are no trust issues between two parties making a transaction. Due to the nature of these smart contracts you will always know that your transaction will have the desired results.

Flaws and possible solutions

The implementation of Bitcoin has shown that it has many flaws, some big examples are scalability and its excessive power consumption (A. De Vries, 2018).
It seems that the PoW algorithm is not only unsuitable for larger scale, as it requires a lot of energy, but also suffers from centralization.

Mining pools

Currently most mining of blocks happens through mining pools. There are collections of miners who combine their computation power to have a better chance of getting a reward. Currently if three of the main mining pools *(Blockchain.info, 2018)* would choose to work together they could reach more than 51% of the whole networks mining capability. This would mean that they could execute a so called “51% attack” which would mean that they could allow or censor malicious transactions without anyone being able to stop them. This gives a few centralized mining corporations all the power.

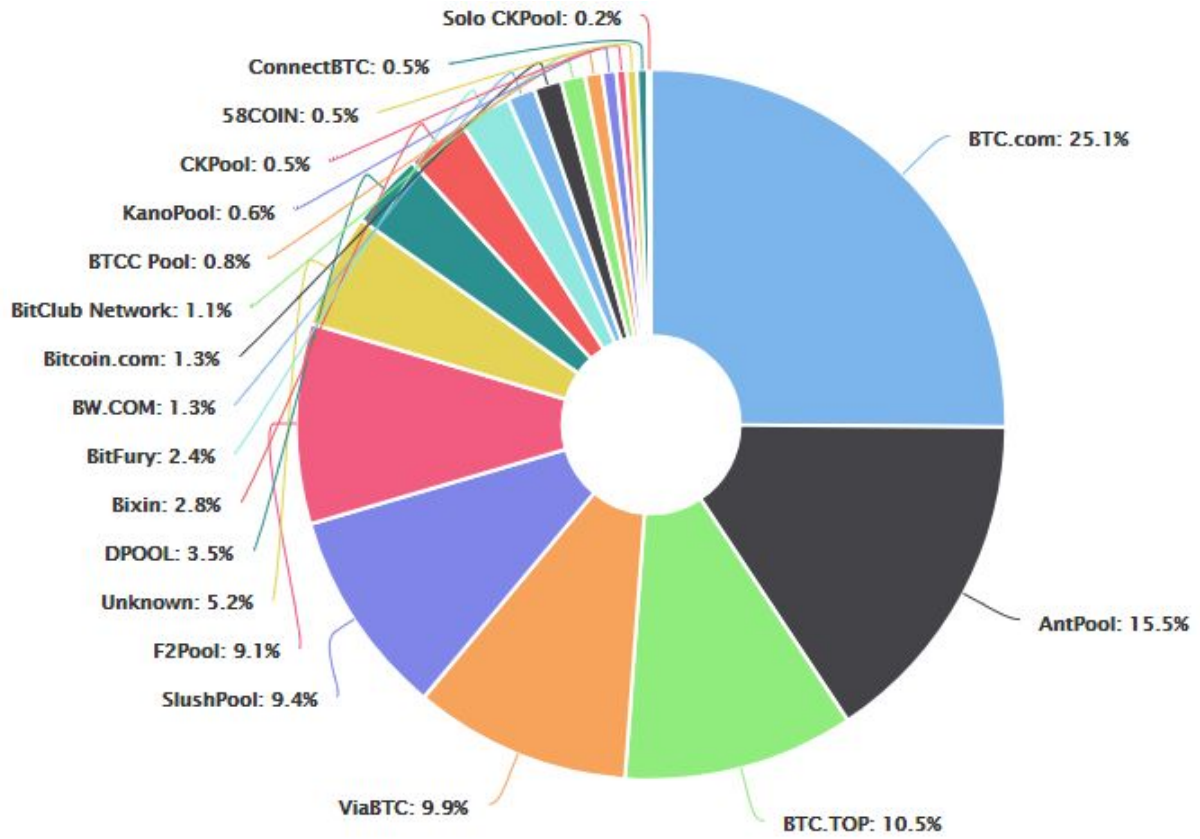


Figure 4: The Bitcoin mining pool computing power distributions (Blockchain.info, 2018)

Bitcoin Scalability

Besides this, bitcoin has also shown to be unscalable (P. Glazer, 2018). During times of heavy use, transaction fees and transaction confirmation times would go up to tremendous heights (Average Confirmation Time, (2018)) due to more transactions being produced than the network can handle.

Proof of Stake

As a result of these flaws Proof of Stake (PoS) was proposed. Instead of proving expenditure of computing power (essentially what PoW is), every miner proves possession of their tokens on the network. By 'staking' these tokens these can be seen as collateral. People who stake their tokens will then be able to verify transactions. They will earn new tokens proportional to their original stake whenever they successfully verify a transaction. This consensus mechanism also solves the 'Nothing at stake' weakness of PoW. In PoS, a tremendous amount of tokens is needed to disrupt the network. Getting such a huge amount of tokens is very expensive. This leads to massive shareholders not being incentivized at all to disrupt the network. This would be because the value of the tokens would drop massively due to public reactions, which would lead to massive monetary losses for these shareholders.

Steem & EOS

As a result of Ethereum's drawbacks and complications mentioned above, there has not been an decentralized application that stood out to non-blockchain savvy people. With Steemit, this is about to change.

Steem

Steemit is a decentralized application, running on its own blockchain called Steem. Steemit is comparable to Reddit to an extent, it is a platform where users can post articles and images. People can then upvote this content. The difference is that the platform is decentralized. This affects the user experience in a couple of ways; the user needs to manage tokens also called Steem. The user's influence on the platform is determined by the amount of Steem the user has staked. Influence is expressed in two ways;

- The amount of 'voting power' a user had
- The amount of posts a user can make per day

When a user makes a post, other users can like this post. The more influence people exert on this post (i.e. the more users like this post), the more tokens this post will generate. These tokens are then paid out to the user making the post. These tokens are, just like most cryptocurrencies, freely traded on cryptocurrency exchanges for any other currency.

Steem makes use of a new decentralized consensus algorithm called Delegated Proof of Stake.

Delegated Proof of Stake

Delegated Proof of Stake (DPoS) is principally similar to Proof of Stake, the main difference is that it is not all users that can produce blocks. Producing blocks is *delegated* to so called block producers. Since there are only 20 (Steem Center, 2018) block producers in Steemit, reaching consensus can be made much easier. This made way for fast improvements in transaction throughput and costs of storage and transactions on such blockchains.

The network remains decentralized by allowing all token holders of a blockchain running DPoS to vote on who the block producers should be. This way, influence and power on the blockchain remains decentralized.

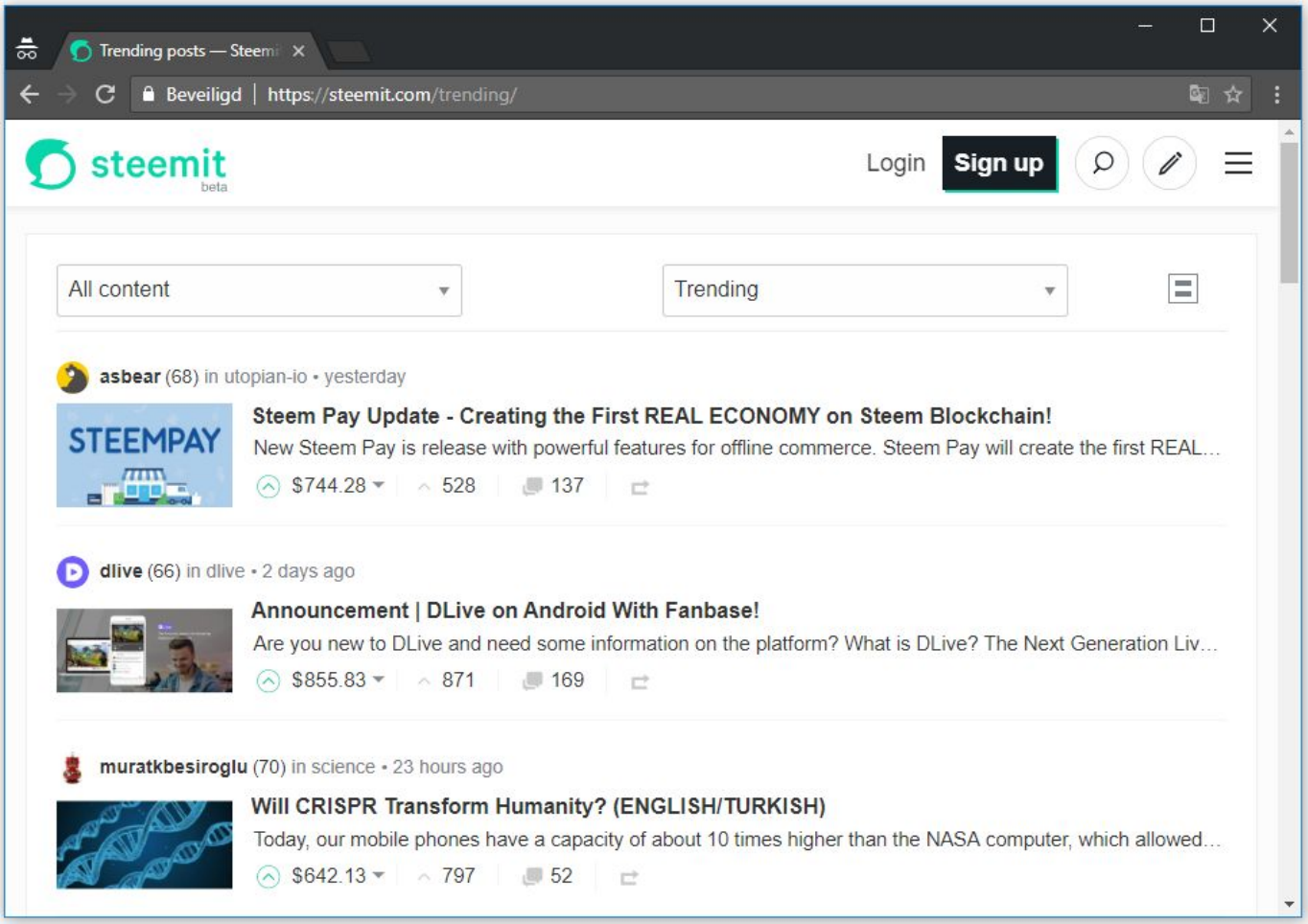


Figure 5: The Steemit website[1]

[1] The Steemit website: http://steemit.com

EOS

After Steem, it became clear that there is a demand for such decentralized applications. EOS is a platform that allows for the creation of decentralized applications similar to Steem in computational capacity.

EOS, along with Steem, brought along many useful features that were much needed. Some of these features include account recovery, permissions systems and usernames.

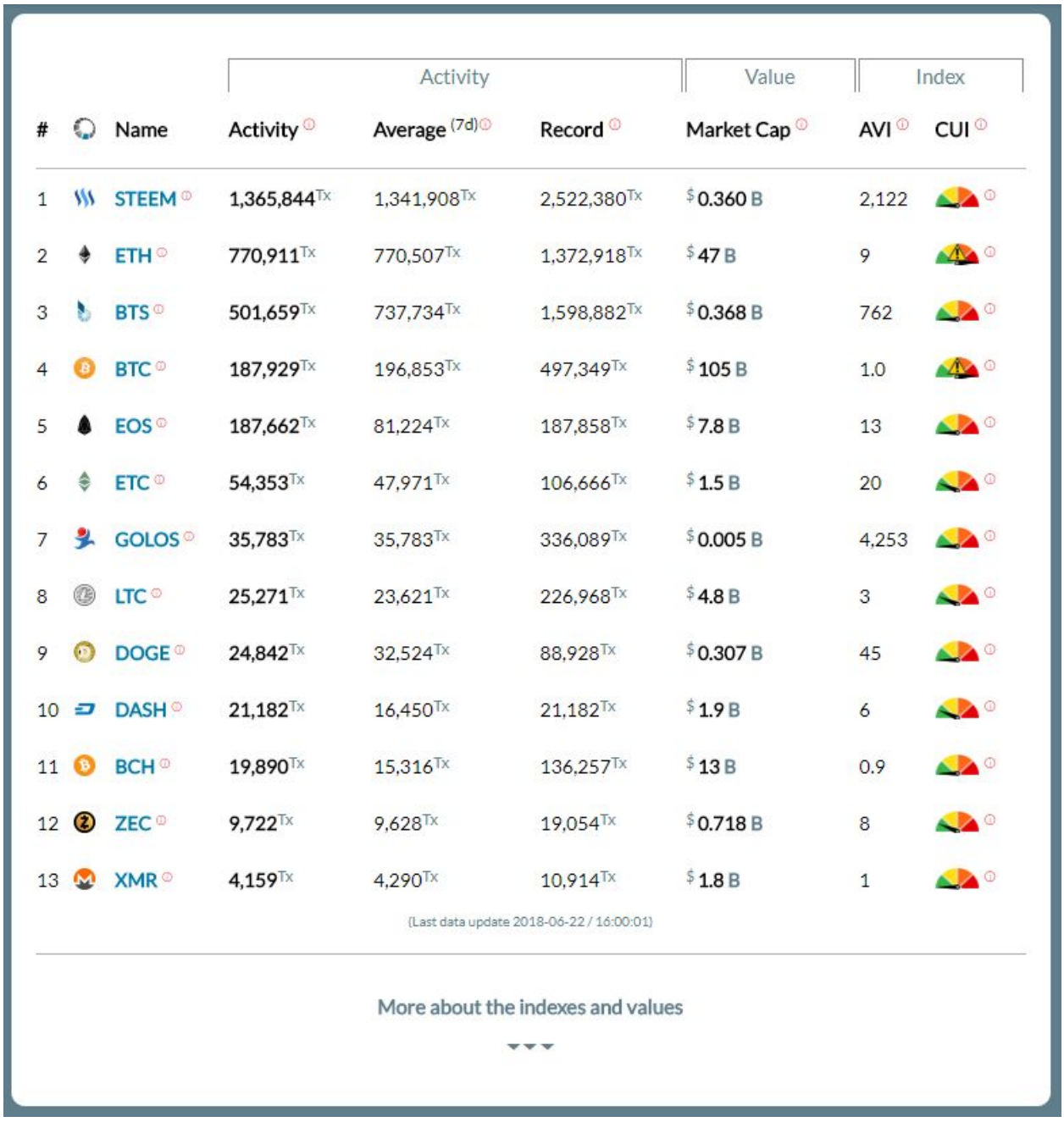
Decentralized applications on EOS run on smart contracts. In principle, EOS smart contracts work in a similar way to Ethereum smart contracts. It consists of action handlers and tables. Any user can call such actions, which in turn can modify, delete or add entries to the tables. The amount of action calls, and the amount of storage any user can utilize, depends on the amount of tokens that user has delegated to bandwidth and storage.

Aside from this, EOS Smart Contracts can perform action on behalf of other users, (given that permission is granted from those users). EOS Smart contracts can also do what is called Deferred Transactions. Deferred transactions are actions that are set to execute after a determined amount of time has passed. It is also possible to set up an interval to perform such actions periodically. These are features that Ethereum does not have.

Current Activity

As we can see from figure 6, the amount of activity on blockchain platforms is widely different from platform to platform. We can see that Steem, which runs a service regular people use, has the highest amount of activity. We can see that Ethereum has the second most amount of activity. It is important to take the utilized capacity into account when assessing this statistic. From the CUI figure in figure 6, we can see that Steem handles this amount of stress with ease, while Ethereum runs at maximum capacity.

EOS is 5th amongst all blockchain platforms, this is impressive considering it has only been released on 14 June 2018. We can see that EOS handles all of these transactions with ease.



			Activity			Value	Index	
#		Name	Activity	Average (7d)	Record	Market Cap	AVI	CUI
1		STEEM	1,365,844 Tx	1,341,908 Tx	2,522,380 Tx	$ 0.360 B	2,122	
2		ETH	770,911 Tx	770,507 Tx	1,372,918 Tx	$ 47 B	9	
3		BTS	501,659 Tx	737,734 Tx	1,598,882 Tx	$ 0.368 B	762	
4		BTC	187,929 Tx	196,853 Tx	497,349 Tx	$ 105 B	1.0	
5		EOS	187,662 Tx	81,224 Tx	187,858 Tx	$ 7.8 B	13	
6		ETC	54,353 Tx	47,971 Tx	106,666 Tx	$ 1.5 B	20	
7		GOLOS	35,783 Tx	35,783 Tx	336,089 Tx	$ 0.005 B	4,253	
8		LTC	25,271 Tx	23,621 Tx	226,968 Tx	$ 4.8 B	3	
9		DOGE	24,842 Tx	32,524 Tx	88,928 Tx	$ 0.307 B	45	
10		DASH	21,182 Tx	16,450 Tx	21,182 Tx	$ 1.9 B	6	
11		BCH	19,890 Tx	15,316 Tx	136,257 Tx	$ 13 B	0.9	
12		ZEC	9,722 Tx	9,628 Tx	19,054 Tx	$ 0.718 B	8	
13		XMR	4,159 Tx	4,290 Tx	10,914 Tx	$ 1.8 B	1	

(Last data update 2018-06-22 / 16:00:01)

More about the indexes and values

Figure 6: A table showing the amount on activity on all mainstream blockchains. Activity is measured in transaction in the last 24 hours. Capacity Utilization Index (CUI): A ratio of blockchain daily activity to total blockchain capacity. (Blocktivity, 22-6-2018)

Game Design Phase

To create a game on the blockchain that is enjoyable to play it is important to take the time to design the game from the bottom up. The game design decisions should not be made arbitrarily. The success of this game also hinges on the quality of the game concept itself. The purpose of this section is to lay out the process of designing the game and to provide reasoning for the design of the game. This process of game design is performed in the following steps:

1. The client specifications are outlined.
2. The creative search space of the game design is reduced according to the client specifications.
3. Game categories will be explored that exist within this reduced search space.
4. Categories will be eliminated and a choice of categories will be made.
5. This category will be explored and a game concept will be chosen.
6. Justify the decisions of the game details.
7. Test the game concept for potential problems.

The BlockSpace team believes that this process should lead to well reasoned game decisions.

Additionally the Game Design document detailing the design can be found in the appendix.

Client Requirements

Fundamental to this project are the requirements the client has laid out. These requirements affect the type of game that can be designed. The requirements are as follows:

1. The BlockSpace Game is to be developed on EOS blockchain.
2. The game must have the potential to be profitable for both the client and the players.
3. The game must have a broad target audience and be accessible to non-blockchain savvy people.
4. The game development must be completed before the end of June.

Search-Space Reduction

In this section, the effects of the client requirements on the design search-space are laid out.

No fast-paced games

The EOS blockchain has 0.5+ seconds of latency, which does not allow for fast paced games. Game genres such as shooters and racing games are not feasible on EOS.

No games relying on asymmetric information

The EOS blockchain cannot secretly provide information that other players cannot see. This makes games that require asymmetric information infeasible. This applies to games such as card games: e.g. poker.

No games relying on randomness

Blockchain is deterministic, any player should be able to get the same results using the same input. This means that randomness in games would be predictable. This causes games relying on randomness to be infeasible. E.g. gambling games.

Game must rely on a token

The profitability of blockchain applications hinges on the use of a custom token currency within the application. In-game these tokens could be used to grant players power and influence within the game. It must be possible to earn these tokens by playing the game. These tokens can then be sold for real money, making the game profitable.

Casual gameplay

The broad target audience makes it hard to design a game that requires a high level of skill or commitment. For a game to become popular with all or most groups of people, it should not have a high level of time commitment, such as World of Warcraft, and should be more casual, like Candy Crush, for example.

No Complex games

Strategy games often involve many decisions and outcomes the players must take into account. Making these games can be quite complicated. Due to time constraints, the game cannot be too complex. Complexity in this context, refers to game mechanics that rely on many variables or much player input with many (predefined) outcomes.

Search-Space Exploration

In order to explore the different directions the game could take, the team organized a meeting with the clients. In this meeting, the team presented the restrictions the game categories must adhere to. The following game categories were the result of this meeting.

Deposit Based Play

One possible game concept we thought of was to have ‘deposit based play’. This means that at the start of a game all players involved will deposit a certain amount of tokens. The players battle against each other to become the victor. After the game has finished the winner of the game will receive all the tokens that have been deposited. This cycle will occur during each game played.

Collaborative Content Creation

During the process of designing the game we found that it could be desirable to have a game that has a community driven approach. The game would have the feel that the community works together on something or competes with each other for something. Either way the game has a feeling of having a big community in which many people play at the same time. One way to create this sense of community is to make the game a collaborative creation game. With this concept everyone works together to create something as a group.

Deathmatch Free For All

Another possible game concept is to have a free for all type game where the player can prepare for a death match based on the amount of tokens/coins they hold. This could mean that the player is stronger/bigger depending on how many tokens they have. However the downside of this is that the player will lose more tokens if the player dies.

Turn Based Strategy

Lastly a turn based strategy game was considered. Here a time based round will be played in which people can perform an action. At the end of each round, the actions performed will be executed. Power in the game could be dependent on the amount of tokens the player holds. Thus making it easier for the player to achieve their actions.

Category Elimination

Here the team discusses the possible issues encountered with each concept proposed. This will be used to determine the best choice of concept category.

Deposit Based Play

This concept suffers from the problem that the game might be considered gambling. This might limit its reach because of its potential conflict with gambling legislations.

Deathmatch Free For All

This concept may suffer from the problem of requiring real time based elements to be interesting. Real time gameplay is difficult to implement when using a blockchain based implementation. The goal with such a game type would perhaps be to create a game concept that hides the delay the blockchain would produce to make it feel as part of the gameplay.

Turn based strategy

This idea seems too complex to execute in a short timespan as strategy based games require more testing and would require a longer development span. During the development process you would have to take into account all the intricate steps a player can make while playing the game. This will make the design process more difficult and take more time.

Collaborative Content Creation

From the game concepts we have decided that Collaborative content creation suits us best. As it does not suffer from the same issues as the other categories, while having the potential to retain players and not become too complex. The player will add complexity to the game, not the game itself. The main drawback of this type of game is that when there is no good community to support the game, the game could potentially become boring and die off. Another potential drawback would be that the gameplay is very simplistic. You draw and you vote, that's it. However, we also kind of see this as an advantage. The game becomes easier to explain to other people, and this could make the game more appealing to casual players.

Search-Space Exploitation

In this section we will come to a conclusion on what game concept best fits our needs and how it will be structured. This game concept will be created from the Collaborative Content Creation category. Some possible concepts for a game of this type are listed below.

Collaborative Music Creation

One concept to explore in this space is a collaborative music creation game. This would involve players cooperating to create a piece of music. They could add tunes on top of each other or next to each other. They could then for example vote on which piece of the song they like. This way the bad tunes get eliminated and a song is created. This could potentially be very promising, however it would take a lot of effort to create a platform for different sounds and beats for the players to combine and harmonize with.

Collaborative Sculpting

The concept of sculpting and building in a collaborative way has already been explored in various games. An example of a game in the style of 3d building would be Minecraft. This concept would involve players working together to build a structure in the game. This concept, although possible, is not feasible in the time frame given for the project. Also it would demand more from the players, making the game less casual, which would go against our objective.

Collaborative Drawing

The last concept that we found could be used is a collaborative drawing game. Collaborative drawing would be somewhat similar to the sculpting concept in the sense that you are also building something. Rather than in 3d and building structures the players would be creating drawings. This concept could be less time intensive for the players, while also keeping the creative element. As a result, it is a more casual experience.

Our Preference

Our preference for creating a game goes to the drawing game concept. This concept is simple to create and feasible to develop within a short timespan. As such, the game complies better with our needs and requirements for the project than the other concepts.

Refined Game Concept

As we have chosen to go for a drawing game, we now elaborate on the detailed design decisions of the game concept.

Drawing Mechanism

Every player gets to paint any arbitrary pixel on this canvas with a color taken from either the default colors, or a bought color. A player will not be able to draw a pixel with the same color as the pixel itself, since this would be could make people spam overwrite pixels such that they 'own' that pixel and receive its reward.

There are two main drawing methods: drawing by staking and drawing by spending.
In drawing by staking, the player will be able to place a pixels for free, however there is an interval in between the pixel placements. This interval decreases as the player accumulates more tokens. This limitation is put in place so that players cannot spam pixels for free. In drawing by spending, players can spend one token, to place a pixel instantly.

This would enable faster, and possible riskier gameplay, while also providing the system with a 'token sink'. This token sink would compensate for the amount of tokens created for rewards, to bring more stability to the token's value.

Tokens

It is important to decide on a correct total supply of tokens such that the value of a token correctly matches with the real monetary value. As an example, you would not want 0.001th of a token to be worth one dollar as this would be annoying for players to comprehend and work with. We want to balance the token amount with the expected market cap of the tokens as well as with how much the rewards will be worth and the additional purchases will cost. Estimating a total market cap of $1,000,000 a total supply of 100,000,000 would be a good amount. This will make 100 tokens be worth $1.

Giving out these rewards would mean that tokens would have to be created to pay out the rewards. As the game tokens would preferably have a max yearly inflation rate of 5%, the daily reward amount should be $(\sqrt[365.25]{1.05} - 1) \cdot 100,000,000 \approx 13500$ tokens with a total supply of 100,000,000 tokens. Over time, this amount will increase as there will be more tokens everyday. Because tokens may also be spend into the token sink, the amount of tokens may reach an equilibrium.

The cooldown on placing a token will be declining depending on the tokens you have staked as well as the duration of a game iteration. The cooldown period for each player (in seconds) will be calculated with the following formula: $\frac{1}{c \cdot (at^2+b) \cdot s}$ where c is the max amount of clicks per second in total we would want for the infrastructure, s is the players tokens staked fraction of all tokens staked, t is a decimal number from 0 to 1 representing the time elapsed during the game iteration and a and b is determined by solving for our prefered end time interval at the end of the game iteration. This rate does not give incentive for players to spread their tokens over multiple accounts instead of keeping them on one account as the clicks per second (= cooldown $^{-1}$) time will increase linearly depending on your tokens staked. This rate will also make sure that if the deadline of a game iteration is approaching the cooldown time will quickly increase. This is done by dividing by $at^2 + b$, where a is chosen negative and b such that the whole expression remains positive. The expression $at^2 + b$ now degrades as t approaches 1. Dividing by this expression raises the cooldown later in the game. If the cooldown time does not increase it would give people a big incentive to only play at the end of an iteration. For more motivations and elaboration for the cooldown formula, see section A of the appendix.

Colour Choices

We want to give players enough colors to use such that they can be creative and have fun creating. At the same time we also want to keep the drawing process simple and do not want to clutter the screen with too many colors to choose from. Creating too much choice of what colors to use can make the process uncomfortable for players that are quite casual. We want to give the player 16 base colors. The player could then later be able to buy additional colors in exchange for game tokens.

Canvas Size

In order to simply visualize the drawing board, while at the same time providing simple atomic actions to the player, the drawing board will consist of a canvas with $m \cdot n$ pixels. The exact size of the drawing board may depend on the technical limitations and the feedback given during the testing of the game, however its starting dimensions will be $1000 \cdot 1000$ pixels as this is the initial size used by other similar games.

Collaboration & Competitiveness

Since a player will have little influence on the board by playing the game on his own, players will have to cooperate with their community to create bigger drawings within a reasonable timespan.

In order to make the painting mechanic more competitive on its own, anyone can draw wherever they like. Paintings are drawn in parallel. Players can choose to draw on his ‘own project’ just as easily as they can draw on someone else’s project. Players can find all drawings of the current competition round on the canvas. To choose a drawing to vote on, the player can browse the voting interface of the application. Here they will find all drawing submissions of that competition.

Competition

An art competition will take place within the game. This adds an extra aspect on top of the game to make the player earn rewards. In this competition the players get to vote on the best drawing made during that competition round. Those who participated in the creation of the winning picture during this competition will receive game tokens as a reward.

Game Iterations

Each iteration of the competition will last for a set period of time. This period will likely be 24 hours long. This period length will give most people around the world the time to participate and draw and would limit the effect timezones have. To give players the opportunity to always be able to draw we want to always have a canvas available to draw on. Since the voting period for the competition requires a static canvas, we have to take a snapshot of all pixels for people to vote on. This makes it such that new players joining the game for the first time will immediately be able to draw and will not be stuck waiting for a voting period to end. As opposed to keeping the drawing from last period and continuing on it, the new canvas at the start of each voting period will be a clean white canvas. We would like to promote a fast paced drawing experience where new drawings are constantly created. Cleaning the canvas after every period will incentivize creating new drawing and will avoid having the same drawing win each time. The down side of this would be that it would feel bad for the player to see his paint he worked on disappear after the voting on it has ended.

Voting Mechanism

We would like players to vote on the drawing they find the best. However we do not want to limit the player to only one vote. We want it to feel more like a upvote system, where players will be able to upvote multiple drawings that they feel are good. The risk of making people only able to have one vote is that they will vote for one painting only so that another painting will not win, effectively creating a sort of two party system.
Having players able to upvote drawing will make it a fun experience for the player. They will be able to look through the submissions and like whatever they feel deserve it.

A problem that can maybe be encountered is that people may only upvote larger drawings as these would have more detail in them, resulting in some smaller drawing not being able to win a competition. As a solution we could scale the votes' weight with the size of the drawing submission. This way the smaller pictures also could have a chance. However since we feel that this would make the process unnecessarily more complex, for the time being we will keep to absolute voting weights.

Potential Problems

We see that each game concept and implementation brings its own unique problems. In this section we discuss some aspects of the game that we foresee may become a problem. We also cover what potential solutions we have found for these problems.

Bot Creation

We want to keep the game playable for everyone. This means we want to avoid having players simply create drawing bots that spam the field with pixels. For this we implement a timer for each action such that you have to wait a specific amount of time before placing your next pixel. The waiting time can be based on the amount of tokens the player holds. This would still allow people to make bots, however it would diminish their unfair advantage.

Last Minute Destruction

To increase competitiveness, the competition only considers the state of the drawing at the deadline for the competition. This could lead to players waiting for the last possible moment to ruin other paintings. Doing this would increase their chance of winning. This is not a healthy playstyle and should be discouraged. To discourage this, we think that the delays for placing pixels should increase as the deadline approaches.

Small Community

A big problem with community based games is: What happens to the game if there is no community supporting it? Once the community is gone will the game stop being playable? We believe that a reward system of tokens for a competition will help mediate this problem. Since this will still give people a reason to keep playing even if the community is small. Having fewer will give you a better chance to win the competition and thus earn more tokens.

Hijacking Painting Rewards

Let's say a group of players creates a painting. After this, some random player creates the submission to the competition for this painting. If the rewards were distributed to the person who made the submission of the winning painting, he would hijack all the results. For this reason, the rewards are exclusively distributed to the contributors of the painting. In this example, the group of players gets all the rewards, while the random player gets nothing.

Overlapping Voting Submissions

As the game allows players to draw anywhere, there are no predetermined sections that the players can submit for voting. This means that it is possible for two players to submit the same or very similar sections of the canvas. This could be solved by having the players stake some tokens in order to submit a drawing. Having overlapping submissions does not necessarily lead to problems. It does not lead to unfair advantages to paintings or players. For this reason, a proper solution might unnecessarily complicate the game.

Increasing Wealth Inequality

The idea of the game is to give players with more tokens more drawing power. This could result in the rich getting richer and the poor stay poor. This can be somewhat mitigated by the community aspect of the game. The rich and the poor can cooperate in a drawing, making it so they both benefit from it. After all, the rewards are not distributed according to wealth, they are distributed according to contribution to great drawings.

If a rich player also happens to be a great artist then he will always outgrow the competition, however we believe that this growth would be justified.
It is important to remember that the rewards are not distributed equally amongst the contributors of the winning paintings. They are distributed to the winners according to the amount of pixels that they contributed to the winning painting.

Lack of Voting Incentive

Currently there is no direct reward for voting on paintings. A concern could be that this might lead to players not voting, or only voting on themselves. One solution to this problem, could be to reward voting on winning paintings. However, we believe that this new mechanic could also be abused. Players could for example vote based on popularity instead of artistic talent. Rewarding players for every vote they place would also not be beneficial, as they would be incentivized to spam votes.

We do not see a solution to this, however it is also unsure whether this really is an issue at all. If a drawing is genuinely good, there should always be enough people who like it. Not every player has their own exclusive painting group, we think that most players will just look around the map and contribute on things they like. As such we suspect that the attitude of most players will be neutral enough, as to not break the integrity of the voting system.

Inappropriate Drawings

As the game will not have a central controlling authority, there is the possibility of players drawing inappropriate things on the game canvas. This is a problem, as the game is supposed to be for everyone and this might cause some groups to be marginalized. We have faith that this problem will usually be solved by the community itself. We believe that most people are not racist or fascist, and will attempt to correct such behaviour if it were to arise.

Software Overview

The software of the project can be divided into two different sections, the Smart Contract and the Client. In this section we will explain how each of these sections works, and what their purpose is.

Smart Contract

The smart contract contains all the game logic, and its role is to act as a ‘server’ for the client. It is structured into actions which the client can call to change the state of the blockchain, and tables which can be used to read the state of the blockchain.

The smart contract was coded in C++, and it comprised two files, the .cpp file, which contained all the actions and assisting functions, and the .hpp file which contained all the tables used by the contract.

When building the contract, the C++ files resulted in the following files, which will be used by the blockchain. The .abi file, which is a JSON-based description on how to convert the database and the actions to and from JSON, and the .wast file, which is a Web Assembly file that contains the contract, and is the only format the blockchain accepts.

The smart contract consisted of 7 actions and 8 tables. Below a snippet of a simplified action of the smart contract can be seen. This action is responsible for allowing users to vote on a submission. It does so by adding an entry in the vote table associating the user with the submission they are voting on, if that entry does not exist already. Then it adds the users vote power to the submissions’ total vote count.

```
// @abi action
// Action called by users to vote on a submission
void votesub(const account_name user,
                    const uint64_t id) {
    // Check user authorization
    require_auth(user);

    // Get the previous date
    uint16_t yesterday = getToday() - 1;

    // Primary key search on submission table for id.
    auto sub_itr = submissions.find(id);

    // Check if the submission is valid. If not, fail.
    eosio_assert(sub_itr != submissions.end(), "The submission doesn't exist");
    eosio_assert(sub_itr->day == yesterday, "The submission is from the wrong day");

    // Secondary key on vote table for user sorted in ascending order.
    auto votes_sort_on_user = votes.get_index<N(user)>();
    auto vote_itr = votes_sort_on_user.lower_bound(user);

    // Skip all votes made by the user that do not match this submission.
    while (vote_itr != votes_sort_on_user.end() &&
            vote_itr->user == user &&
           (vote_itr->id != id || vote_itr->day != yesterday)) {
       ++vote_itr;
    }

    // Check if the iterator is not pointing to the vote of the submission. If it is, fail.
    eosio_assert(vote_itr == votes_sort_on_user.end() || vote_itr->user != user, "Already upvoted");

    // Add a vote to the vote table, the contract pays for it.
    votes.emplace(_self, [&](auto &v) {
        v.vote_key = votes.available_primary_key();
        v.user = user;
        v.id = id;
        v.day = yesterday;
    });

    // Add the users vote power to the submissions vote counter.
    submissions.modify(sub_itr, _self, [&](auto &s) {
       s.vote_count += (uint64_t) votePower(user);
    });
}
```

Data structure

In order for the game to work as intended, we needed to store the following:

- Information about the user account.
- The pixels drawn by each user.
- The submissions made by each user.
- The votes made by each user for each submission.
- The winning submissions waiting for processing.
- The users participating on the winning submissions.

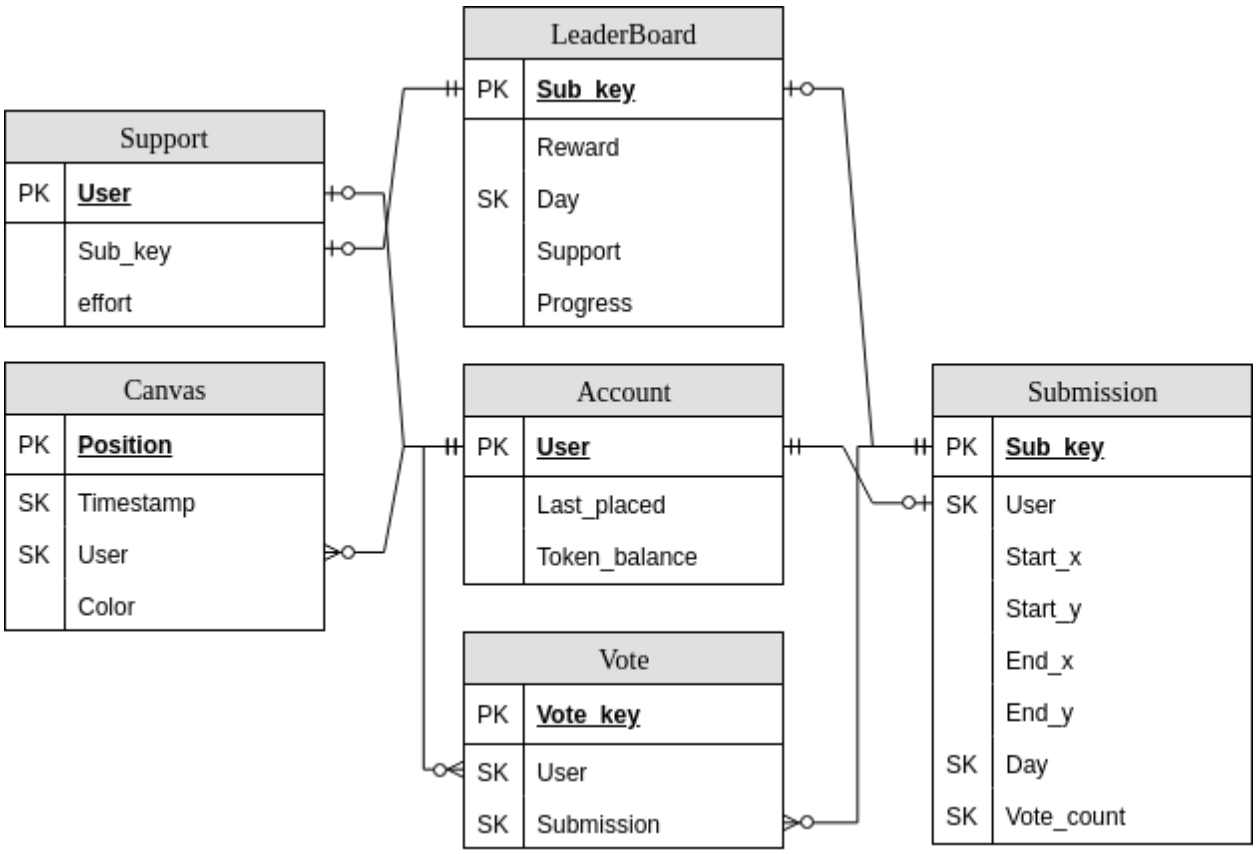


Figure 7: Smart contract data structure

Due to the way EOS works, data storage can only be done in tables, somewhat similar to a relational database. Additionally, any field or table name can only be up to 12 characters long. The data structure as seen in figure 7 was implemented. In the following sections we shall describe its components.

Canvas

The main component of the game is the canvas where players can paint, this canvas is needed to store the position, color, time of painting and the player who painted each pixel.

A core concept of the game was the competition, in which players would paint and submit entries on one day and vote on them on the day after and, on the third day, the winners would be displayed. This caused a problem, which was that players would be idle for two days in between painting sessions, to solve this we created 3 canvases, each running the game at a different stage, so when canvas 1 is painting, canvas 2 is voting and canvas 3 is displaying the winners.

Account

Although it is not necessary for us to store the user accounts due to the nature of EOS, we still needed to store extra information to attach to these accounts in our game. This information consists of in-game token balance and the last time a user placed a pixel. To do this, the team created a table containing this extra information.

Submission

For the competition, players need to submit their drawings which we need to store. But as there is a large possibility of submissions overlapping, storing the pixels for each submission is inefficient. Thus we only store which area of the canvas each submission covers and then use the final canvas of that day to display the submission.

Vote

In order to prevent players from voting twice on a single submission, it is needed to store which player voted for which submission. This is done by the vote table, which, when a player votes for a submission, creates a entry linking the player to the submission, which can only be removed by removing the vote or deleting the submission.

LeaderBoard

When the voting period is over, the smart contract needs to give out the rewards to each user participating in the winning submissions in proportion to how much they contributed.

Due to the nature of EOS, the smart contract can only handle small amounts of data each transaction before it times out. Additionally EOS is not able to completely calculate the participation in a submission with more than 100 painted pixels in a single transaction. In order to solve this we distribute the calculating process, and the leaderboard table holds the progress on this process.

The information the Leaderboard table contains is;

- the key of the winning submissions;
- the reward each of them will be getting;
- the date matching the submission it is pointing to to ensure it has the correct one;
- the total support by the players for the submission;
- the progress on calculating the support;

Support

While the Leaderboard table stores the information about the winning submissions, the support table, keeps track of how much support each user has given to each submission being calculated at the moment. This is done in order to give each user an appropriate amount of the reward for that submission.

Client

The Blockchain is open for everyone with an account to communicate with. However there is no game without a user interface which makes the game more fun to play and also suits itself for people who are not particularly technically literate. The client's main function is to interpret the data that it retrieves from the blockchain and presents it to the users in a clear and easy to understand way. The client makes users able to communicate and send messages to the game's smart contract without needing to understand what is going on in the background.

Frameworks

In order to implement the web interface on time and not go out of the scope of the project, the following frameworks were used;

Game Engine

The game was created as a web based game. This makes it accessible to a large audience. To put less focus on how to make a game on the web, and to focus more on blockchain aspects as well as game design, it is wise to make use of a game engine. For this, we used Phaser (Phaser.io, 2018), which is a game engine built for browser games. This engine was used to handle the visualization and input of the game elements such as e.g. creating the game canvas, placing pixels on the game canvas, zooming the canvas, voting on drawings made etc.

Phaser provides a great amount of documentation and tutorials on developing a game. The Phaser site provides a large array of feature examples, which contain code snippets with explanations of how the features can be created. It has a large community and has grown to be one of the biggest and most popular javascript based game engines.

Web Frameworks

For Javascript ECMAScript 6 (ES6) was used. ES6 provides many advantages over traditional Javascript and has introduced many features that were previously only possible through use of obtuse workarounds. To deal with asynchronous calls it provides its Promises structure. To ease the creation of anonymous functions it introduced lambda functions. Additionally ES6 also provides the creation of classes and also gives an elegant way to import and export modules. As the game is web based, html and css was used to create static elements in the UI.

Blockchain API

In order to communicate with the blockchain through the application some kind of API is required. A nice javascript based API is provided by eosjs (EOSIO/eosjs, 2018). This Framework gives easy to use functions to get information from the EOS blockchain such as account information, smart contract tables and more. When a function in eosjs is called and the response is retrieved, a Promise is returned that can be used to filter for the information you need. It also provides functions to make it easy to make transactions. In figure 8 an example is given of how a transaction is made in eosjs. This transaction calls a function of the smart contract to paint a pixel on the canvas. If a transaction fail, an error is sent.

```
24      drawPixel(usr, xParam, yParam, col) {
25        return this.eos.transaction({
26          actions: [
27            {
28              account: 'blockspace',
29              name: 'draw',
30              authorization: [{
31                actor: usr,
32                permission: 'active',
33              }],
34              data: {
35                user: usr,
36                color: col,
37                x: xParam,
38                y: yParam,
39              },
40            },
41          ],
42        });
```

Figure 8: A snippet of a transaction carrying a 'draw' action to the EOS Blockchain

Coding Style

As we like to have a clear and consistent coding style, the airbnb (airbnb/javascript, 2018) style guide was used. In order to adhere to the chosen style it is good to have a linting tool. The linting tool will recognize errors in the code style and will suggest solutions to fix them. ESLint is a widely used linting tool for ES6 to ensure code quality. In combination with a good text editor such as WebStorm or Atom this gives an easy way to fix code styling errors.

Responsive Mobile Application

It is good to have an application that is responsive and suits itself well for current mobile devices. For this reason the application is packaged into a Progressive Web App (PWA). Progressive Web Apps gives the user the option to add a website to his app home screen and make it feel as though it is a standalone application. The app has a clear full screen view and is able to load resources when offline. This way the game feels like a standalone application. This makes it such that a larger audience is reached while eliminating the work necessary to develop for different platforms.

Development process

This section gives an overview of the tools used to improve the development process as well as what the working environment was like.

Work Environment

To work on the project the client, YX Impact Innovations, invited us to work at their new office at Weena 200 in Rotterdam. They gave us our own desks to work at as well as providing a lunch. We worked on the project every day from 9 to 5. This gave us a good environment to be productive during the day, increasing the quality of our work.

Scrum and User Stories

For the Development of the application we adopted a lightweight scrum approach. This means that we created issues for features at the start of each week to guide us in the sprint for that week. Due to experience from projects in the past we found that it would be counterproductive to focus too much attention on Scrum documentation such as retrospectives and extensive backlogs. We found that it was easier to discuss sprint in person and document them as issues on a GitHub Board rather than extensively following the proper details of the SCRUM procedure. Another consideration that we took into account was the fact that we only had three sprints, setting up a proper SCRUM procedure would have given too much overhead.

Code base and Version Control

For version control we made use of git. For our code base we made use of Github. Github provides many good features such as Pull Requests for code review. We made sure that no code was merged, unless it was reviewed by at least one team member by forcing at least one review per pull request. We also made the master branch protected such that there was no possibility of pushing unreviewed code to the master branch. We also used the Github Projects feature to create issues for features and to keep track of our progress for each sprint.

Continuous Integration

To make sure the project always builds, we used Continuous Integration. For this we used Travis CI. Travis CI made sure that no GitHub Pull Request was merged to the master branch, before the code properly compiled and had no styling errors with ESlint.

Code Structuring Preparation

To ensure good structure of the codebase where methods and classes are not too complex or too long, we made sure to plan the structure ahead of time. Before starting coding we outlined what the general structure of the application should look like. After this, we created the empty classes and empty functions such that only the implementation was left to do. Each function contained comments on how the implementation should entail, and how the communication with other modules should be.

Development environment

As the official EOS network was not up during the time of development, part of the development process was focused on setting up a private test network on a dedicated server in which we could deploy the smart contract.

Communication

Due to possible problems with privacy regulations we were advised not to use Slack for communication. Because of this, we used Mattermost to communicate with the TU Coach and the BEP Coordinators. For in-team communication we made use of a WhatsApp group and a private Mattermost channel as well as a lot of in person discussion. There were times, especially during the development sprints, where communication with the TU Coach and/or the client was not needed. We were focused on creating the software that was previously agreed on in the weeks prior. During this time, there was very little communication with the TU Coach. In the future, we should let the TU Coach know beforehand.

Since we worked daily on the project at the office of our client, it was easy to communicate with the client and receive direct feedback. Additionally we also used a WhatsApp group to communicate with the client. The team also received a lot of useful feedback from both the coach and the client when it came to documents related the the Game Design.

Game Experience Testing

Testing is an important part of every part of the project, not just the software. To make sure we were making the right choices, it was also important to test the concept of the game itself.

To this end, we gathered as many friends and acquaintances as possible. With them we performed two main play testing phases. From these test phases, we got a lot of feedback that we have used to improve the game as a whole. These two testing phases are laid out in this section. We settled on two test phases as that was most appropriate given the timespan of the development of the project.

Testing phase one: Pixel Art Competition Prototype

For this testing phase, we used a very early pixel art prototype we created. The amount of test subjects we had for this phase was 15 people. These 15 people consisted of close friends of the team. Most of the test subjects were not particularly familiar with Blockchain. ⅔ of the test subjects filled in the survey that came along with this testing phase.

This testing phase served three concrete goals. These goals, the execution of the tests, and the results will be explained in this section.

Test Goals

In the first testing phase, we wanted to test the aspects of the game that are not related to blockchain at all. Instead, we wanted to gauge engagement and observe whether people are in favor of the web interface solution.

Goal one: Gauge interest and engagement

The first goal of the first testing phase was to measure the demand for the game we wanted to create. We wanted to find out whether people liked the idea of having a game wherein the main goal was to participate in daily pixel art competitions. We wanted to see for ourselves whether content creation indeed was as engaging as we thought it was when we chose that focus.

Goal two: Test the choice of colors

The second goal of this testing phase was to find out whether the color scheme we chose is appealing.

Goal three: Test the Progressive Web Application Interface

The last goal of this testing phase was to find out what our potential player base thinks about the type of client we chose for the game. We chose to make the client a Progressive Web Application and collected feedback on this.

Test Execution

First, to test a concept with actual users, there needs to be a (simple) prototype of the aspects that we want to test. To this end we made a Progressive Web Application wherein users can paint pixels. To realise the daily competition, we set up a small test group where players would submit the paintings they made that day. At the end of the day we would communicate to the users who the winner of the competition would be. This test group was also used to communicate about the game and to provide feedback on the prototype.

We also set up a survey for the test subjects. This survey consisted of multiple choice questions and is given in the appendix.

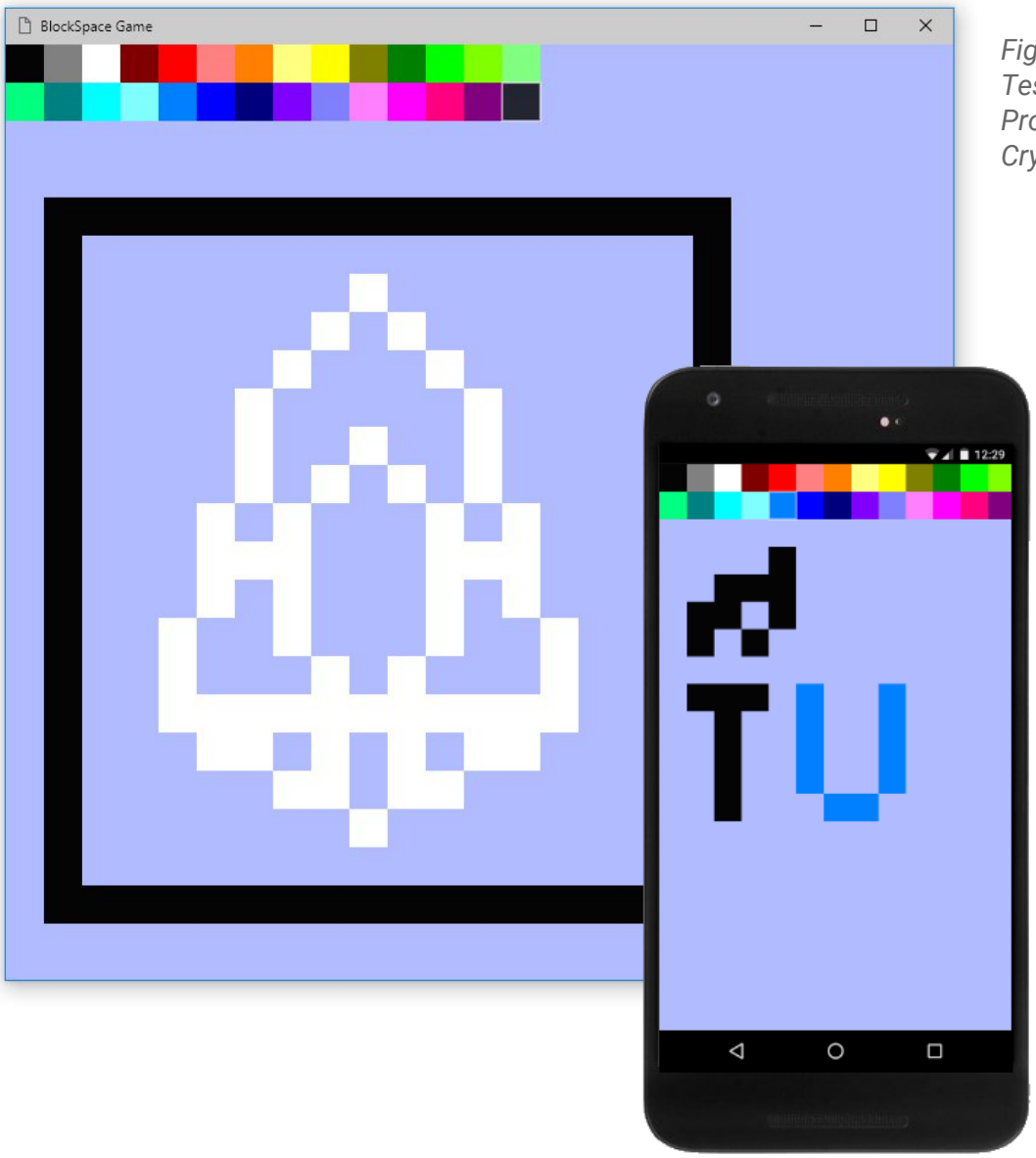


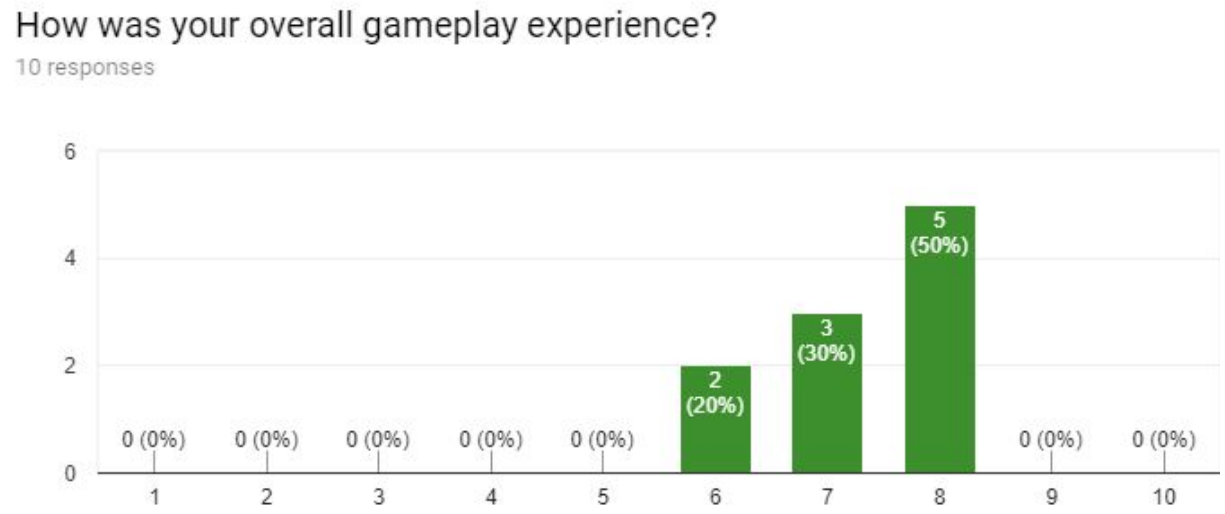
Figure 9: First Testable Early Prototype of CryptoPIX

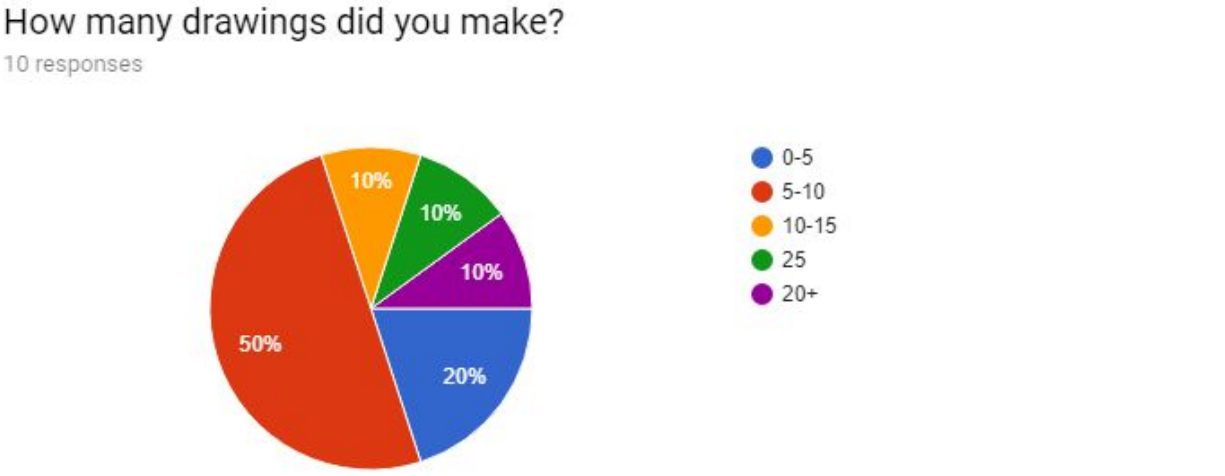
Test Results

The results and feedback of the first test phase are given in two forms. Firstly we used a survey of simple multiple choice questions for the test subjects to fill out. Secondly we took note of the feedback the test subjects gave verbally and in text communications with us.

Survey results

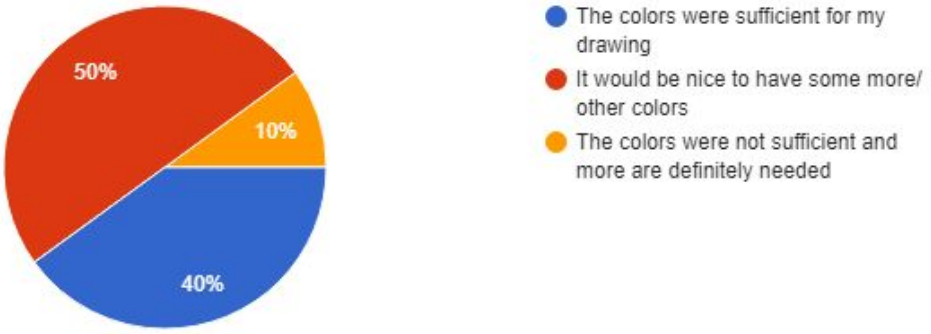
Figures 10-14; The results of the first testing phase.





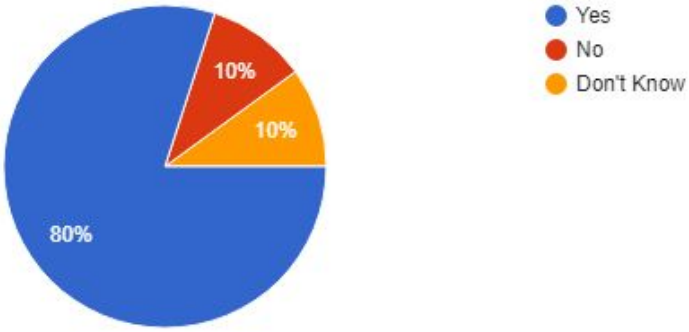
Were the amount of colors good or would you like to have more/other colors

10 responses



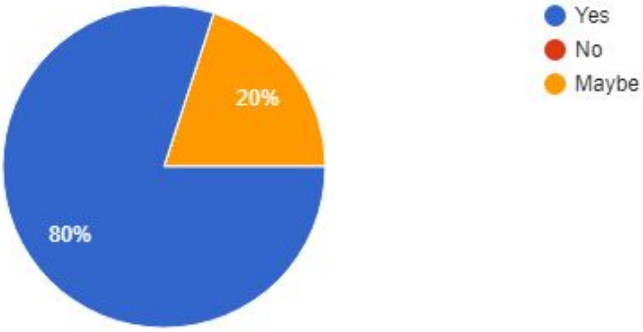
Was drawing pixel art an enjoyable experience as a game?

10 responses



Do you think the daily competition added positively to the game experience?

10 responses



Other feedback

We got many other messages as feedback during this testing phase. Many of the messages were related to missing features of that early prototype. These messages were along the lines of; *Why can we not zoom in/out on the pixels?* Or *Why can we not see what other people are drawing?* These missing features were later added to the prototype for the second testing phase. There was also feedback related to colors; some players felt that certain colors, such as a dark brown were missing.

Most people really liked the web interface. They liked that downloading the application is not really part of the user experience. The test subjects also liked that they were able to play this game using any platform that has a browser. There were however some complaints about the app asking for permissions for sending notifications.

Test conclusions

The first goal was to gauge engagement. The testing subjects we had for this testing phase were overall enthusiastic about the game. We can see in the results that 80% of the test subjects at least sent in 5 submissions, and often more. We can also see that 80% of the test subjects found drawing pixel art to be enjoyable. We concluded that these results show that there is enough engagement and interest for this idea, to pursue the development of the game to the second testing phase.

The second goal was to test whether people liked the selection of colors we used. In the results we can see that 10% of people thought that we should definitely increase the color amount. 50% of people would like to see more colors added to the game. This caused a discussion in the group on whether or not we should add more colors. After careful consideration, we decided that we should not extend the AMSTRAD CPC color palette with other colors now. If we have more specific data on what color additions would benefit the game most, the team could add those colors.

Figure 15: Snippet of the submissions we got during this testing phase.

The third goal was to test whether people liked the Progressive Web Application solution for the game. The only complaint we got was the notification popup request, which is unavoidable using this solution. Because of the lack of other negative feedback and the amount of positive feedback on this web solutions, we decided to go further with the progressive web application client.

Testing phase two: Decentralized Blockchain Game Prototype

For the second testing phase we used the same testing group as in the earlier phase. This way we could track the progress of our game and see how the test groups felt about the implementation of their feedback.

Test Goals

In the second testing phase we aimed at getting an idea of the general gameplay experience for the game prototype. This prototype was our minimum viable product. This prototype had full communication with the blockchain and contained the features that we aimed to implement.The aim was to get feedback on what the game experience was like.

Goal one: Test relation between tokens and gameplay

One of our goals was the look at how the relation between the blockchain elements and the game are perceived. We focused for example on how the addition of game tokens was perceived. This included looking at if the token system was complicated to understand and whether the players were actually incentivized to make use of the system by earning more tokens.

Goal two: Test collaborative gameplay

We also looked at what the addition of collaboration was like. We looked at whether it was rewarding for people to contribute to a larger painting where they could either fix errors in the painting, introduce errors or help to create the painting.

Goal three: Test if the game feels like a regular game

Lastly we also had the goal of gauging whether the game was indistinguishable from a ‘regular’ game.

Test Execution

In order to perform testing with the test group, we first explained to the test subjects what the game is about, and how the game works. Then we gave the test subjects the url for the web application to play the game. Along with the URL, we gave them usernames to try and a private key to login. While playing, we took note of their feedback, and after playing we asked the test participants to fill in a survey.

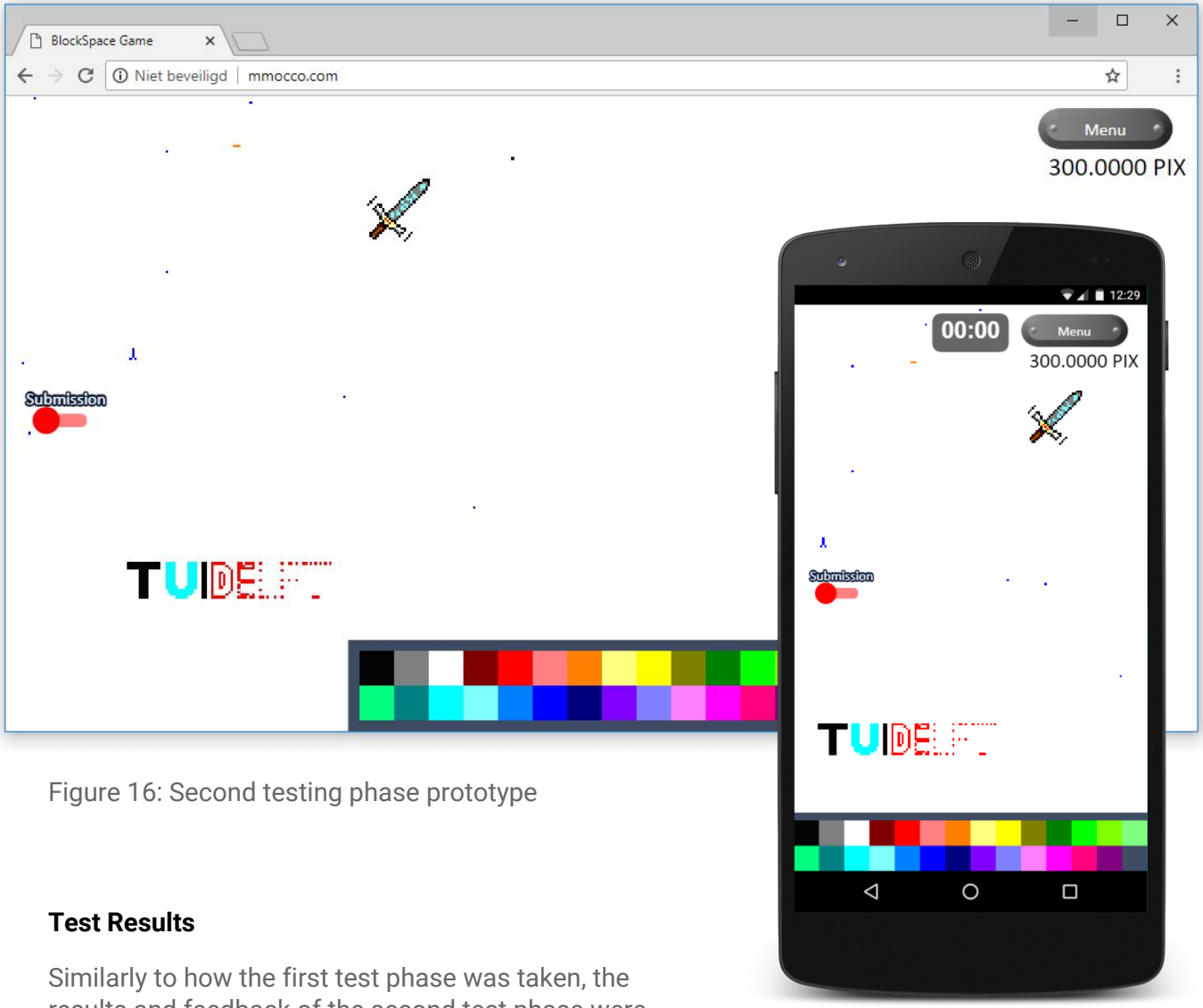


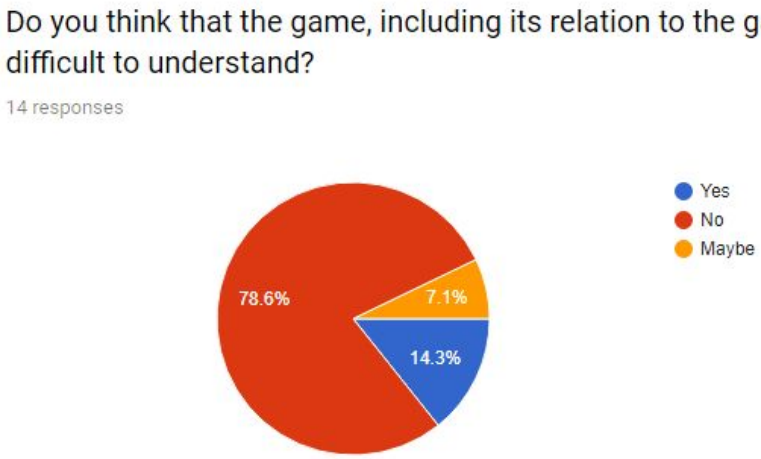
Figure 16: Second testing phase prototype

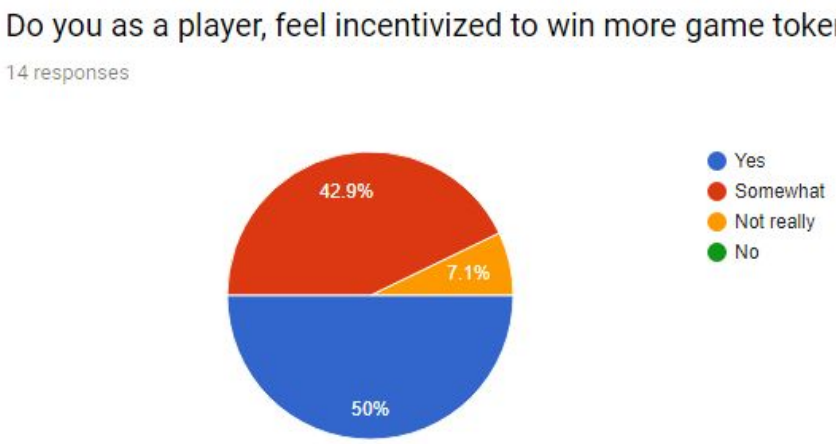
Test Results

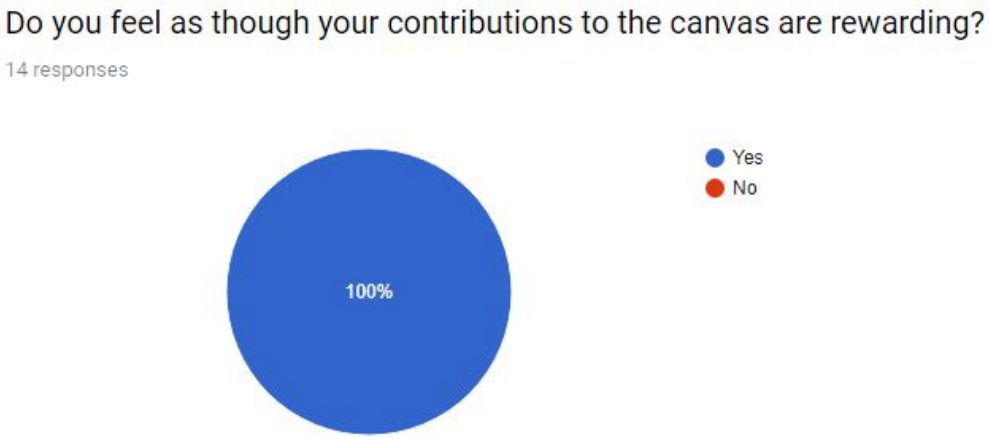
Similarly to how the first test phase was taken, the results and feedback of the second test phase were given in two ways. First by giving a multiple choice survey, and then by taking note of direct, verbal, feedback. In this section, we show these results.

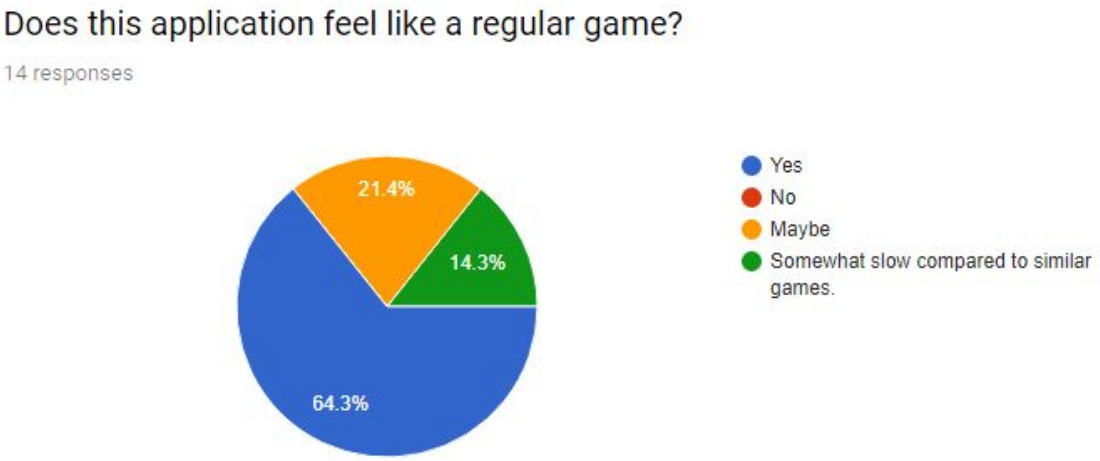
Survey results

Figures 17-20: The results of the second testing phase









Other feedback

In addition to the survey, we also received direct feedback from the test participants. Some feedback that the test subjects gave included, making the system for submitting paintings a more clear process. There was also a suggestion to provide a screen where the submissions made today were directly viewable. We also received feedback on the game still being a bit difficult to easily send to friends/colleagues without giving some amount of initial explanation.

Test Conclusions

From the results for this testing phase, we have made the following conclusion with regards to our testing goals.

From the feedback we received it was shown that the majority of users had little problem with understanding the relation between the gameplay and the tokens in the game. Most of the users felt incentivized to use the token system and the possible rewards to earn new tokens. However from the data we can see, that there is still a little bit of reservation for the participants as to whether they feel incentivized to earn more tokens. This is visible in the results of "*Do you as a player, feel incentivized to earn more game tokens*" where 42.9% of the participants answered with somewhat.

With regard to testing the collaborative drawing experience, we saw that most participants stated that they felt some kind of accomplishment/reward, from contributing to the painting canvas.

Lastly we answer the question: "*Does this application feel like a regular game?*". We see that there is no person that answered this question with 'no'. The majority answered with either 'yes', or some with a reserved 'maybe'. Even though this sample pool for the participants was relatively small, this still gives us an indication that the goal of making the game indistinguishable from regular games is achieved.

Software Improvement Group (SIG) Feedback

As part of the Bachelor End Project, we got the opportunity to send our code to SIG to receive feedback on its quality. In this section we paste the feedback (Received In Dutch) and the way we dealt with and incorporated the feedback.

Feedback Week 5

De code van het systeem scoort 3.2 sterren op ons onderhoudbaarheidsmodel, wat betekent dat de code marktgemiddeld onderhoudbaar is. We zien Unit Interfacing en Unit Size vanwege de lagere deelscores als mogelijke verbeterpunten.

Voor Unit Interfacing wordt er gekeken naar het percentage code in units met een bovengemiddeld aantal parameters. Doorgaans duidt een bovengemiddeld aantal parameters op een gebrek aan abstractie. Daarnaast leidt een groot aantal parameters nogal eens tot verwarring in het aanroepen van de methode en in de meeste gevallen ook tot langere en complexere methoden.

In jullie project zijn er een aantal punten waar coordinaten (startX, startY, endX, endY) als aparte parameters worden doorgegeven. Dat is een typisch voorbeeld waar een parameter-object de code beter leesbaar zou maken. Het lijkt overdreven om voor zo'n klein voorbeeld een object te maken, maar je ziet vaak dat er op lange termijn steeds meer parameters bijkomen, en parameter-objecten zijn een goede manier om de aanroepen van deze methodes duidelijk te houden.

Voor Unit Size wordt er gekeken naar het percentage code dat bovengemiddeld lang is. Het opsplitsen van dit soort methodes in kleinere stukken zorgt ervoor dat elk onderdeel makkelijker te begrijpen, te testen en daardoor eenvoudiger te onderhouden wordt. Binnen de langere methodes in dit systeem, zoals bijvoorbeeld, zijn aparte stukken functionaliteit te vinden welke ge-refactored kunnen worden naar aparte methodes.

In jullie project is createGameBoard() in Paint.js een goed voorbeeld. Alle logica staat nu in één methode, wat op lange termijn een probleem gaat worden als het opbouwen ingewikkelder gaat worden. Er zijn nu twee aparte fases: het bord opbouwen en het toevoegen van event handlers. Door die te scheiden wordt het makkelijker om de delen onafhankelijk van elkaar te testen.

Als laatste nog de opmerking dat er geen (unit)test-code is gevonden in de code-upload. Het is sterk aan te raden om in ieder geval voor de belangrijkste delen van de functionaliteit automatische tests gedefinieerd te hebben om ervoor te zorgen dat eventuele aanpassingen niet voor ongewenst gedrag zorgen.

Over het algemeen is er dus nog wat verbetering mogelijk, hopelijk lukt het om dit tijdens de rest van de ontwikkelfase te realiseren.

Implementation Feedback Week 5

In order to respond to the feedback received on week 5, the code was refactored, improving the organization of the files making the project. Methods became shorter, got less parameters and were divided into sub-methods and the amount of comments increased.

Unfortunately test coverage did not increase due to time constraints and lack of experience using the selected frameworks.

We found that after implementing the feedback, the code became cleaner and easier to read.

Delivered Prototype

This section describes all the features in the Delivered prototype.

In the span of the project, a prototype has been developed for which the following was implemented;

Login screen

The login screen consists of input fields for a username and private key. When the login button is pressed, a public key is generated from the private key. The app uses this to check using the blockchain whether the user exists. If the user is found, the app will transition to the painting screen, if not, a message indicating that the input was invalid appears.

Figure 21 Login screen

Painting Canvas with Color Palette

When the users open the painting canvas, the screen shown in figure 22 will be displayed. It consists of the painting canvas in the background, the color palette at the bottom, the submission toggle on the left and the menu button and the amount of PIX (token) display on the top right.

When a color is chosen on the color palette, and the user clicks the canvas, the pixel that has been clicked will become of that chosen color.

When the menu button is clicked, a pop-up menu will show up allowing the user to go to the voting screen, or logout.

The user can move the canvas around by dragging it, and is able to zoom in and out by pinching the screen (on mobile) or scrolling (on PC).

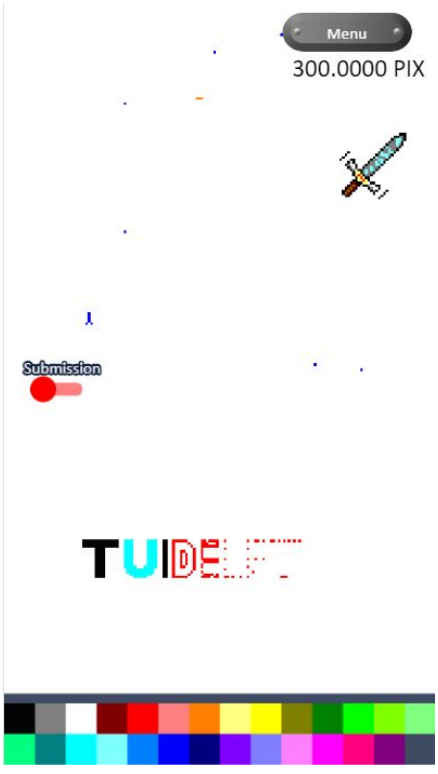


Figure 22 Painting canvas

Submission Creation

When the user clicks the submission toggle while in the painting canvas state, they enter the submission creation mode. This mode has the same layout as the painting layout, except that the color palette is not visible and some helpful text appears in its place. Clicking creates one corner of the green rectangle seen on the right. Clicking a second time will finish the rectangle allowing users to submit the selected area by clicking the newly made rectangle.

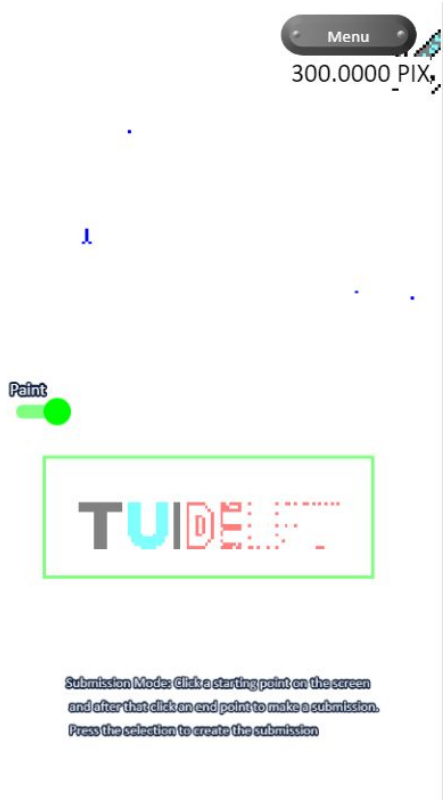


Figure 23 Submission creation

Voting screen

The voting screen consists of a list of submissions and a menu button as seen in figure 24.

The submission list shows all submissions made on the previous day. The user can scroll through all the submissions through the ScrollView. The user can upvote any of these submissions simply by clicking them.

When the menu is opened the user can choose to go to the painting canvas or logout.

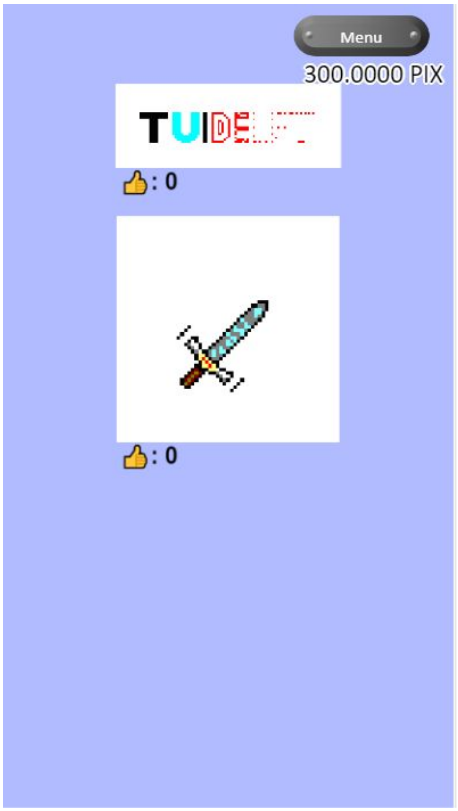


Figure 24 Voting screen

Conclusion

This thesis covers the process of implementing a video game in the EOS blockchain. From it we see that although it has its limitations, it is already possible.

At the start of the project our research question was: "*Is it possible to create a decentralized game that is actually fun and feels like a 'regular' (centralized) game?*"

When it comes to making the game engaging and fun, we see that during the testing process participants showed that they enjoyed playing the game. The participants did not seem to notice a big difference between the gameplay of regular games and our game.

Making the a decentralized game does however require a different mindset during the development and design process. Blockchain provides new features to games, such as token based economies and persistent games which are not dependent on the game developers' support.

More fast paced games built on a decentralized platform may suffer from issues such as transaction times. This currently limits the game design space of decentralized games.

Judging from the results from the testing phases, we have concluded that the EOS blockchain is a suitable technology to build games upon. Older blockchain implementations seem to struggle with performance and scalability while EOS has shown to have improved upon these aspects. Creating games using blockchain technology, from our results, does not necessarily have a negative impact on the gameplay experience. However it does seem to limit the design space when designing games, as blockchain does not suit itself for all game types.

Recommendations & Ethical Discussion

In this section we shall include recommendations on going forward with the project, along with ethical dilemmas that may appear from creating a system such as the one that was built for this thesis.

Recommendations

In this section we would like to give some recommendations to the client and for the further development of the game.

Private key handling

We recommend the team to add the option for using Scatter (EOSEssentials/Scatter, 2018) to handle private keys of players. Scatter is downloadable as a browser extension. Scatter allows users to sign transactions and store data without the need of exposing your private key to the website. We would recommend the addition of this option so that it will be easier for users to play the game while feeling safe about their cryptocurrencies.

Experimental Software

We would advise the team to in the future stay away from very experimental software. It is good to first be sure that something has been tested and well documented. This will reduce the amount of time spent of research how the software works and will leave more time for actual implementation.

Ethical Implications

One of the ethical problems that the team will encounter when further developing and releasing the game, is that people will trade real money for this new in-game token. Due to the free market nature of these tokens, they will most likely be able to be traded on token exchange platforms. Token prices in the crypto space have shown to be very volatile, meaning that they can go up or down very quickly. The decisions the game's development team makes, can have significant impact on the finances of the playerbase. As such, we will try to keep changes to a minimum, we do not want to introduce errors that could potentially make players lose money.

Another ethical problem that we should consider is the possibility of people creating images that are deemed offensive or illegal. This would include hate symbols such as swastikas. Trying to moderate the creation of this on a blockchain application would mean that we would have to introduce moderators to remove such content. However this would mean that, in some sense, there would be a central authority that decides what can be drawn. This would go against a great principle of decentralization, as such we do not consider this a good solution. As stated in the design phase, we believe players will try to moderate each other to remove such hate speech or illegal content.

During the development process, we also stumbled upon a dilemma on whether our servers should store private keys for the players. Storing private keys could increase the playability for players greatly as they would be able to have a simpler login experience where they are not required to enter their private key. However when storing private keys, we would face all kinds of ethical problems such as being responsible for the users' funds, and having the danger of getting hacked which could cause the keys to be leaked. Because of the problems this brings with it, we decided on letting the user store their own private keys.

Further Development

Since we accomplished our goal to; *"Create a decentralized game that is actually fun and feels like a 'regular' (centralized) game."*, we are planning to continue the development on the game after the conclusion of the Bachelor End Project.

Here we discuss some of the further features that the team believes should be added, along with what other future work should be done.

Create Automated Software Tests for game components

It is important to make sure that game components and their communications always perform as they should. Because of this, we would like to add extensive automated unit, integration and end-to-end tests for the game software.

Token Sale

The team is planning on holding a token sale for the in-game tokens. Before the beginning of the token sale the team will participate in a short marketing period.

Improve the User Interface/Experience

The team would like to improve upon the graphical elements of the game and make it more appealing to other players. The current game experience can be improved on, by having more explanatory elements such as tutorials or FAQs.

Improve Registration Process

The team would like to add an easy process for people to register an account through Google third party authentication services. The account created through this, will also be granted some initial tokens. In this way the free to play experience for players is good enough to enjoy the game, without feeling too limited or held back.

Reflection

Looking back on the project we can see that we have learned a lot. We have learned how to develop smart contracts for applications on the blockchain. We have learned about properly setting up a project for a web based application. We have learned a lot about important concepts in web development. We have learned how to produce software for an external client.

We have also learned that we should be careful with the use of experimental software that is not well tested or documented. Even though some experimental software might be very useful, it can also prove to be very difficult to use or set up if there is little to no documentation.

We have learned that doing software development in a setting where the group members can talk to each other face to face, makes for a very efficient and productive work environment.

We are happy to say that we believe this thesis adequately represents the effort that we invested into the bachelor end project.

REFERENCES

1. airbnb/javascript. (2018). GitHub. Retrieved 22 June 2018, from: https://github.com/airbnb/javascript
2. Asolo, B. (2018). Delegated Proof-of-Stake (DPoS) Explained. Retrieved 19 June 2018, from https://www.mycryptopedia.com/delegated-proof-stake-dpos-explained
3. Average Confirmation Time. (2018). Blockchain.info. Retrieved 21 June 2018, from https://blockchain.info/charts/avg-confirmation-time?timespan=180days
4. Baghla, S. (2017). Origin of Bitcoin: A brief history from 2008 crisis to present times. Retrieved 21 June 2018, from https://analyticsindiamag.com/origin-bitcoin-brief-history/
5. Bardinelli, J., & More, R. (2018). 10 Promising Cryptocurrencies to Watch in Q2 2018 - Invest In Blockchain. Invest In Blockchain. Retrieved 21 June 2018, from https://www.investinblockchain.com/10-promising-cryptocurrencies-q2-2018
6. BBC News, CryptoKitties cripple Ethereum blockchain. (2017).. Retrieved 25 June 2018, from https://www.bbc.com/news/technology-42237162
7. Block'tivity. (2018). Blocktivity.info. Retrieved 22 June 2018, from http://blocktivity.info/
8. Bowles, N. (2017). CryptoKitties, Explained ... Mostly. Nytimes.com. Retrieved 19 June 2018, from https://www.nytimes.com/2017/12/28/style/cryptokitties-want-a-blockchain-snuggle.html
9. Crosby, M., Pattanayak, P., Verma, S., & Kalyanaraman, V. (2016). Blockchain technology: Beyond bitcoin. Applied Innovation, 2, 6-10. http://scet.berkeley.edu/wp-content/uploads/AIR-2016-Blockchain.pdf
10. Dai, W. (1998). B-money. Retrieved 21 June 2018, from http://www.weidai.com/bmoney.txt
11. de Vries, A. (2018). Bitcoin's Growing Energy Problem. Retrieved 21 June 2018, from https://blog.ethereum.org/2015/07/30/ethereum-launches/
12. EOSIO/eosjs GitHub. (2018).. Retrieved 22 June 2018, from https://github.com/EOSIO/eosjs
13. GitHub. (2018). EOSEssentials/Scatter. Retrieved 25 June 2018, from https://github.com/EOSEssentials/Scatter.
14. Glazer, P. (2018). The Bitcoin Scaling Debate: Context, Proposed Solutions, and the Future. Retrieved 21 June 2018, from

https://hackernoon.com/the-bitcoin-scaling-debate-context-proposed-solutions-and-the-future-579f9373e24b

15. Hashrate Distribution. (2018). Blockchain.info. Retrieved 21 June 2018, from https://blockchain.info/pools

16. Hashrate Distribution. (2018). Blockchain.info. Retrieved 22 June 2018, from https://blockchain.info/pools

17. Kith, N. (2018). EOS (EOS) – Rise to the top unstoppable with the release of EOSIO Dawn 3.0. Retrieved 19 June 2018, from https://cryptorecorder.com/2018/04/14/eos-eos-rise-to-the-top-unstoppable-with-the-release-of-eosio-dawn-3-0/

18. Mappster, C. (2017). Why EOS is Better Than Ethereum & Bitcoin. Retrieved 19 June 2018, from https://steemit.com/eos/@chiefmappster/why-eos-is-better-than-ethereum-and-bitcoin

19. Martin, L. (2017). Star Wars Battlefront 2 release date: True cost of DLC REVEALED, as fans turn on EA. Express.co.uk. Retrieved 19 June 2018, from https://www.express.co.uk/entertainment/gaming/879026/Star-Wars-Battlefront-2-release-date-DLC-EA-Credits-Reddit

20. Morris, D. (2017). Bitcoin Hits a New Record High, But Stops Short of $20,000. Retrieved 19 June 2018, from http://fortune.com/2017/12/17/bitcoin-record-high-short-of-20000/

21. Nakamoto, S. (2009). Bitcoin: A Peer-to-Peer Electronic Cash System. Retrieved 21 June 2018, from https://bitcoin.org/bitcoin.pdf

22. Ofek, E. Richardson, M. (2003). DotCom Mania: The Rise and Fall of Internet Stock Prices. Retrieved 21 June 2018, from http://pages.stern.nyu.edu/~eofek/DotComMania_JF_Final.pdf

23. Phaser.io. (2018). Phaser - The fast, fun and free HTML5 Game Framework. Retrieved 22 June 2018, from http://phaser.io/

24. Steem Witness - Steem Center. (2018). Steem.center. Retrieved 22 June 2018, from https://www.steem.center/index.php?title=Steem_Witness

25. Taggart, M. (2017). Itty Bitty Kitties Are Killing Ethereum. Retrieved 21 June 2018, from https://medium.com/@michaelx777/little-bitty-kitties-are-killing-ethereum-a1253b18b501

APPENDIX

A. Pixel Placement Cooldown Calculations

The cooldown is measured in seconds. Generally speaking, we want that more tokens mean a lower cooldown. Another way of expressing cooldown is by `allowed number clicks per second'. Cooldown can then be derived with: $cd = (cl/sec)^{-1}$ (with $cd =$ cooldown, $cl =$ click, $sec =$ second). We want that cl/sec to be linearly with a player's stake. With s the fraction of stake of a player, this gives motivation for: $cl/sec = [\cdots] \cdot s$ where $[\cdots]$ represents other contributing factors described below.

Define c to be the average total number of clicks per second over one game iteration. This constant will be dependent on the amount of EOS staked in the contract. If we choose $cl/sec = [\cdots] \cdot c \cdot s$ and assume $[\cdots]$ to be roughly 1, we now have a fair way to distribute all of the available clicks per second.

As a last factor, we want the cooldown to increase near the end of the game. This corresponds with a decrease of cl/sec. We now have $cl/sec = f(t) \cdot c \cdot s$ with f a function dependent on the time t, measured between 0 and 1 where 0 represents the start of the game and 1 the end. What we want for f is to decrease over time. We also want f to have an average value of 1. This is because we do not want to increase or decrease the total number of clicks in a single game iteration, we only want to redistribute the clicks. We have chosen f to be of the form $f(t) = at^2 + b$ because parabolas are easy to calculate with and with properly chosen values for a and b, it will produce a nice, smooth decreasing slope.

How do we choose a and b? Assume that at the end of the game, we want the amount of clicks per second to be a fraction d of the average number of clicks per second our contract is able to handle. This gives: $f(1) = d \Rightarrow a \cdot 1^2 + b = d \Rightarrow a + b = d$. We also know that the average value of f should be 1. This gives:

$$\frac{\int_0^1 f(t)dt}{1-0} = 1 \Rightarrow \int_0^1 at^2 + b\,dt = 1 \Rightarrow \left[\tfrac{1}{3}at^3 + bt\right]_{t=0}^{1} = 1 \Rightarrow \tfrac{1}{3}a + b = 1$$

We now have two linear equation with two unknowns:

(1) $\frac{1}{3}a + b = 1$ and (2) $a + b = d$. Now, we subtract (1)-(2): $\frac{-2}{3}a = 1 - d \Rightarrow a = \frac{-3}{2}(1-d)$. This gives an expression for a. Because $a + b = d$, we know $b = d - a = d - \frac{-3}{2}(1-d) = \frac{3}{2} - \frac{1}{2}d$.

We found that $d = \frac{1}{2}$ is a good candidate. This gives us $a = \frac{-3}{4}$ and $b = \frac{5}{4}$.

B. Original Project Description

EOS Blockchain Space Domination Game

Project Description

Blockchain

The uses of blockchain technology are currently mostly limited to; financial contracts and tokens / coins with a market value. The Ethereum blockchain enabled developers to create simple decentralized applications (DApps). However, Ethereum is limited by transaction throughput and data processing power. It is also inconvenient for regular users. The EOS blockchain solves these issues, and enables developers to create a broader range of DApps. This new terrain of blockchain concepts is largely unexplored. YX Impact Innovations seeks to be one of the first companies to successfully create a fluent feeling game on the blockchain. This game should be playable by regular people, and should look and feel indistinguishable from their centralized counterparts.

Visualization

An important aspect of the game is the 3D universe, in which the user will explore a variety of randomly generated planets. This means that three dimensional visualisation techniques will be a challenging part of the project.

Game concept

The game itself can be best described as a strategic space conquering, exploration game. The game's objective, is to conquer a multitude of valuable planets and protect them against other real life players. As with most Blockchain currencies; the tokens used to purchase digital assets in this universe, should have a real life market-value. This allows players to make (or lose) real money by playing the game.

Challenges

- Creating smart contracts on the blockchain to run game logic and handle data storage;
- Creating a web interface the user can interact with to play the game;
- Visualizing aspects of the game in 3D to use in the web interface

The EOS platform is currently experimental technology. Since its main blockchain is not yet launched, you will be developing on EOS's test blockchain, which is under heavy development. Working with experimental and new technology will also pose a challenge.

Potential research question

- Is (the EOS) blockchain suitable to create games in?

Company Description

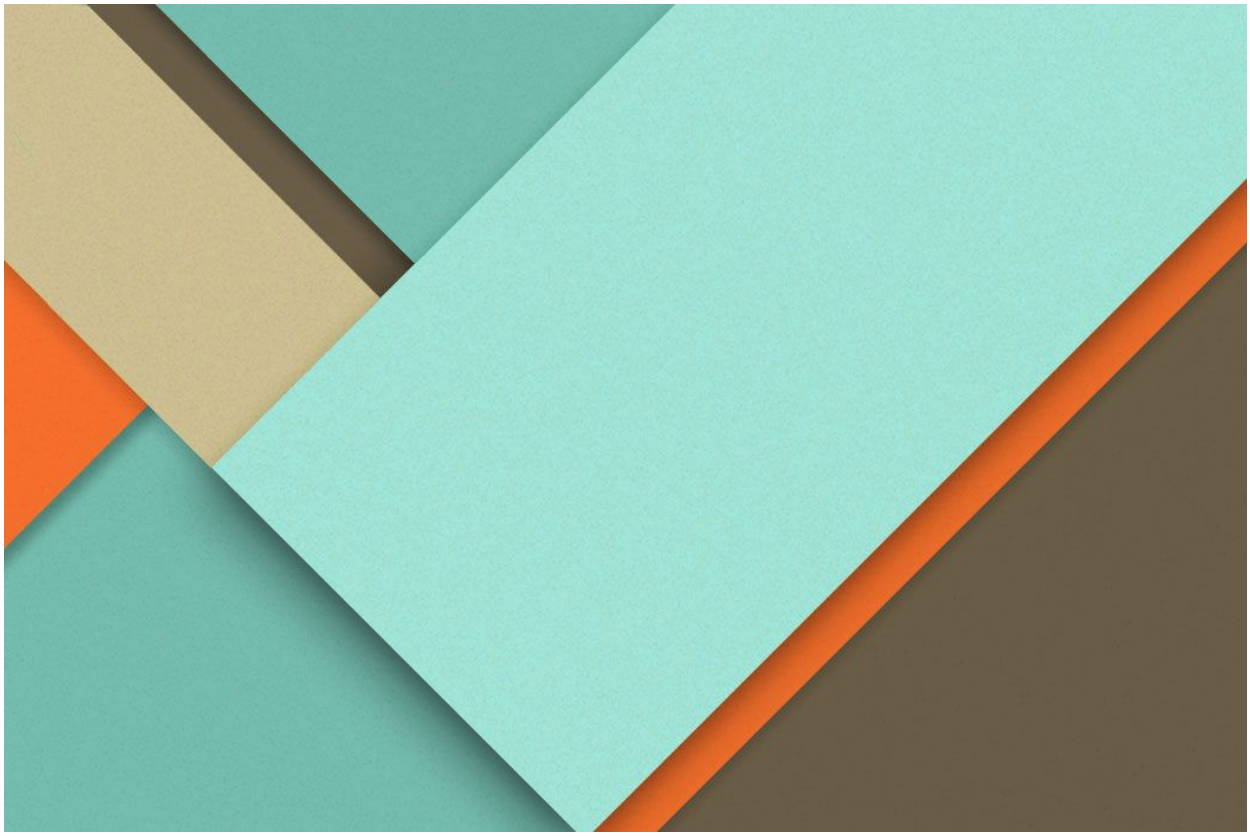
YX Impact innovations is a Dutch startup based in Rotterdam that develops innovative concepts with impact. The company aims to solve problems combining creative minds with state-of-the-art techniques. The company was founded under the name of Easycollect Services in 2015, when they introduced a collection box that supports donations by contactless payments. They soon became the market leader in contactless donations in The Netherlands, as major Charity Foundations started using the innovative collection box. They are currently developing several new concepts, among them a mobile application for debt prevention, an online comparison platform for retailers and a decentralized game on the blockchain.

Additional information

- An Agile SCRUM approach will be followed;
    - The students will be part of a self managing team, guided by a Product Owner from the company
    - Students will participate in daily standups and other regular SCRUM meetings
- Students are free to decide on the type of solution and development tools;
- Students will work on-site on our brand new designed Innovation Floor;

Students can qualify to continue developing the application further after the project/MVP has concluded

C. Research Report



Research Report

What is the best approach to building the BlockSpace blockchain game?

BlockSpace

#------ Kees Fani

#------ Manuel Ferreira

#------ Pavel Hoogland

Table of Contents

Introduction

The aim of this report is to give a comprehensive summary of the research phase of the BlockSpace Bachelor Project. Here the approach taken for finding an optimal solution for the project problem is discussed. Discussed are the choices of technology and methods to be used. The report contains the choices made and the motivations for these choices.

An overview of the development environments used, as well as the structure of the software is given.

Client

YX Impact innovations is a Dutch startup based in Rotterdam that develops innovative concepts that have an impact. The company aims to solve problems combining creative minds with state-of-the-art techniques. The company was founded under the name of Easycollect[6] Services in 2015, when they introduced a collection box that supports donations by contactless payments.

They soon became the market leader in contactless donations in The Netherlands, as major Charity Foundations started using the innovative collection box. They are currently developing several concepts, among them a mobile application for debt prevention, an online comparison platform for retailers and a decentralized game on the blockchain.

Figure 1: Model of Easycollect's collecting box

Brief Project Description

For this project the client has requested a game to be developed, that is to be built on top of the EOS[2] blockchain platform. The client wishes to find out whether the EOS platform is suitable for the purpose of Game Development and whether the aspects that Blockchain technology provide over the traditional architectures are advantageous in the context of a game.

The client sees Blockchain as one of the greatest recent innovations, that might enable them to develop new (or better) solutions to social problems, that could not have been

solved before. This project is about gaining knowledge about the subject, while also working in such a way that the game can actually be launched and be profitable.

Client Requirements

There are some requirements the client has given for the game project. The requirements for the development of the game are:

1. The BlockSpace game is to be developed on EOS blockchain.
2. The game must be easily accessible to non-blockchain savvy people.
3. The game must have the potential to be profitable for all stakeholders.
4. The game must have a broad target audience, giving it a greater potential to become popular.

The research done for the game concept design can be found in the *Game Decisions Reasoning* document.

Blockchain Justification

Blockchain has recently gained much popularity because of its growing market value and use cases. The latter is what interests us most. Blockchain solutions are great for replacing intermediaries[22] and running services in a decentralized manner. Think of services like Uber, AirBnB or even financial institutions responsible for transactions between parties. This project should show that this principle can also be applied to the gaming domain.

In this section the research findings are shown on whether and how blockchain could be a viable choice for video games.

Decentralized Application Logic

Decentralizing application logic is done using so called 'Smart Contracts'. Smart contracts consist of two parts; a database and functions that can manipulate this database. These functions can be called by any user or other smart contract.

Figure 2: Schematic Smart contract showing functions and storage

Application logic for will reside in these functions. The end-users are billed for the execution of the application logic and storage space used.

A common practice in decentralized applications is implementing a token, which gives users of the smart contract more influence on the platform. More influence could mean that the token holders have more privileges, or more voting power in polls or elections within the application.

Reduced Cost

Decentralized solutions have the potential to reduce cost for both end-users and developers. This is because the end-users do not have to pay a fee to an intermediary, while the developer will not have to pay server upkeep costs. Given that the blockchain chosen facilitates cheap transactions and storage, the cost of applications that traditionally need intermediaries should be greatly diminished.

Resilience

Centralized online games have two main issues with regards to resilience. Its first issue is its resilience against local calamities, such as power shortage in the game servers. The second issue is the performance of the game servers in times of increased load or traffic. Blockchain technology solves the first issue by removing the vulnerability of having a single point of failure. Blockchain technology solves the second issue by delegating more of the network's computational power to applications that are in need of it. This delegation of computational capacity is only a viable solution if the blockchain network chosen has enough computational capacity. The choice of blockchain technology will be motivated in further sections.

End-Users are Beneficiaries

In traditional centralized applications, the company behind the application and its investors are the only shareholders. These shareholders have their own interests, which do not necessarily line up with those of the end-users. Companies can make decisions that make the service worse and its clients unhappy for the sole purpose of making more money.

Decentralized applications (DApps) are different in this regard. In such solutions, there is no central authority making decisions. The interests of both the company and the shareholders can be aligned through having a token based model where both the users and the company hold tokens on the platform. Since in DApps, the users are shareholders, everyone's interests are aligned. An increase in token value is beneficial for all parties involved. Decisions that make end-users unhappy will only decrease the value of the tokens. There is no more conflict of interest in this regard, the end-users are the beneficiaries.

Scalability

In normal multiplayer games, there are usually one or more servers in charge of connecting the players to each other, and, as one machine has a limited computing capacity, this creates a need for upgrading or getting new physical or cloud servers to handle the extra load when more players join.

By using blockchain, the need for a centralized server is removed and the computational power demand is moved to the users, making it so that the available computational power is increased every time a new user joins.

Possible Drawbacks

At this point, not every traditional game is possible to be built on blockchain. Games that need fast paced interaction and processing of user actions, such as shooters, are not suitable. This is the case because every blockchain modification has a delay associated with it. Blocks, which contain transactions, need time to be successfully appended to its blockchain, the so called 'blocktime'. These blocktimes are usually in the order of seconds to minutes, in the case of EOS it is approximately 0.5 seconds. As opposed to the 30 to 60 updates per second needed for fast paced online games.

Blockchain technology has overhead associated with it, in terms of modifications done to data needed to run the games. As a result, games that do not need online connectivity would only be slowed down by the use of blockchain technology, without receiving any of its benefits.

The blockchain is transparent, every user can read what is stored in the blockchain. Games that rely on confidential or private information are not recommended. Even though data can be encrypted before being put on the blockchain, this data will always persist. On top of that, applications running on the blockchain would not be able to decrypt the data without leaking the private key.

Randomness is currently not feasible when using a blockchain, as blockchains are deterministic. This means that games that heavily rely on randomness such as gambling games and card games such as HearthStone are not suitable for using blockchain technology. Some workarounds for this randomness problem have been proposed[20]. However these still suffer from predictability and are not appropriate for games that require complete randomness.

EOS Blockchain

By now, a plethora of different blockchain platforms and environments have been created. This has lead to a vast amount of choices at the disposal of the application developer. Some examples of blockchains that enable development of decentralized applications are Steem[18], NEO[14], Lisk[12]. The most popular blockchain used for decentralized applications is the Ethereum blockchain[21]. Many decentralized applications have already been built on the Ethereum blockchain, some of which proven to be quite successful[5]. However these applications have also shown the limitations of the Ethereum blockchain. A popular application (Crypto Kitties) was at one moment taking around 10% of the blockchain transaction power causing the blockchain to be clogged up[2].

Several new up and coming blockchains (sometimes referred to as Blockchain 3.0) try to solve the problems encountered with various current iterations of the blockchain (Blockchain 2.0). One of the candidates that has gained a big following and popularity is EOS. The creators behind EOS have claimed that it can have the ability to reach millions of transactions per second[10]. EOS aims to solve several problems that other blockchains have shown to have. The EOS platform also allows for users to have more privileges and advantages on an application depending on the amount of tokens they hold on the platform. Development of applications on EOS is also possible in C++, which is a well known language, whereas Ethereum is programmed in their own language; Solidity.

EOS' high transaction performance and feature-rich DApp platform has convinced us that EOS is an excellent choice to build the BlockSpace game on.

Development Process

The BEP requires that the development team performs well in a collaborative environment. Collaboration is vital in meeting project requirements in an efficient manner. Having a good development process is essential to having as little problems as possible during the project.

In this section the development environments and processes used are explained. Here the choices made will be motivated, explained and the advantages will be assessed.

Agile

A well known[4] software development approach for enabling this is *Agile Software Development*. The agile approach puts focus on having incremental and iterative changes of the product. Herein, a short time frame is set for producing solutions for some determined requirements. After this timespan, the solutions will be analyzed such that in the next

production cycle, any problems encountered can be resolved. Another focus is the collaborative aspect. Usually a daily meeting is involved where the team members discuss what has been accomplished, what problems were encountered, and what the plan for the next day is.

SCRUM

An approach that implements the principles of the Agile development method is SCRUM. SCRUM implements an iterative approach by having several consecutive sprints. Each sprint is planned at the start of the sprint. During the sprint planning requirements are taken from the product backlog and added to the sprint backlog to be planned and assigned to the team members. Every day a daily standup is performed to discuss the progress of the sprint. At the end of the sprint a Sprint Retrospective will be performed to assess the progress made during the sprint and to address any problems encountered as well as providing improvements for the next sprint.

SCRUM has been shown[7] to be a very efficient method for producing quality software. The SCRUM process gives flexibility for the product owner to make changes to the product during development as well as giving developers a clear schedule to adhere to. By using SCRUM the developers can detect problems in their work process early on and adapt accordingly.

BlockSpace Approach

As SCRUM has been shown to be one of the best approaches to developing software it will also be the main development approach of the BlockSpace BEP project. The project will follow the main aspects of SCRUM. The aim is not to literally follow the SCRUM methodology, but more to give a structure to the workflow. The project will only take the best aspects of SCRUM to reduce time spent on overhead such as making a formal planning and retrospectives. The project will follow a lightweight form where a planning will be made and retrospectives will be created but less focus will be put on producing the appropriate documentation. For the purpose of making the SCRUM backlog with user stories as well as assigning the tasks, GitHub project boards will be used. The Retrospectives will be shown in a short document that briefly discusses any problems encountered for each sprint. Focusing too much on documentation has the tendency to take up a lot of time in a small development team, where there is no dedicated and experienced SCRUM master.

Software Overview

Selecting the correct software solution aspects is a crucial step in the development of any software problem. Making the right choices in this area has great impact on the effectiveness of the software. This project consists of two main parts. One of the parts is the distributed game logic, which will reside on the EOS blockchain. The second part is a way for the user to interact with the blockchain (Game Interaction Solution). The current possibilities for APIs to communicate with the blockchain are limited. The EOS development team have made a javascript based API[9] that provides this link between the application and the blockchain. This javascript API suits itself best for development of web based applications. For this reason the Game Interaction Solution will be made as a website.

In this section an overview will be given of how the two software components will be structured, and how the communication between them will be performed.

EOS Environment

The EOS platform development is still in its early stages. At the time of writing, only recent Linux distributions and Apple computers are supported. Since none of the team members own Apple computers, the project will be developed on Linux distributions such as Ubuntu. The team will be using a docker image distributed by the EOS development team, in order to ensure that every team member runs the same software development versions.

Game logic, and thus smart contracts, shall be built using C++. EOS uses Web Assembly[8] internally and at the time of writing, only C++ code can be compiled to Web Assembly EOS can understand.

Game Interaction Solution

As explained in the intro of this section the game is to be run on web-based platforms. This web application should feel like a proper game app to the end-user. A good candidate to accomplish this, is to make a Progressive Web App[11] (PWA). This web application can be used in a mobile device as if it was a normal app. A Progressive Web App gives users the ability to simply take a website and create an app for that site on their mobile home screen. When they launch this app it will feel as a standalone application instead of a website.

In order to develop a good looking game on the web, it is wise to use a library that handles most of the game elements. One of the best options to accomplish this is the Phaser3 Library. This library has a lot of community support, as well as many developers constantly contributing to its development.

In order to properly write code, it is also good to have a feature-rich code editor. For this, the preference of the BlockSpace team goes to Webstorm editor. Webstorm is an IDE that provides useful functionalities such as debugging, source control support, as well as integrating nicely with the npm package manager.

Since blockchain applications are completely transparent and decentralized, creating applications that interact with the game logic can be done by anyone. There are no explicit rights needed from BlockSpace to accomplish this. This would mean that, even if the current development team stops supporting the website, someone else could create an interaction system for the game logic. The game logic in the smart contract can only be manipulated by its own smart contract, while a way to interact with the blockchain game can be created by anyone.

In order to connect with the blockchain and execute transactions the eosjs library[9] will be used. The BlockSpace team will be using EcmaScript 6 because of its Promise[17] feature. This will make asynchronous code easier to manage.

Quality & Testing

To make sure that software runs properly in many different situations and on different machines, it must be properly tested. To have properly working software we want to make sure that each layer (see figure 3) in the software works well on its own and in combination with other layers.

Code Quality

Good code quality is of great importance for the longevity of the development cycle, as it facilitates easier future updates and readability. For Javascript several coding styles exist. Some good options are Google Javascript style, Javascript Standard style or AirBnB Javascript style. AirBnB Javascript style is one of the most popular and covers most aspects of the language. Because of this the project will be using the AirBnB Coding Style Guide[1]. On top of this, to help ensure that code quality is of a high standard, ESLint will be used. ESLint is a tool that ensures that the code written will adhere to the standards we have decided on. EOS Smart contracts do not yet have any code styling preferences defined, so the BlockSpace team will use their best judgement here.

Unit Tests

To ensure that components of the code work correctly, Unit testing will have to occur on both the web application, as well as the EOS smart contracts. The EOS.IO team made an experimental unit testing framework. As this is the only viable testing framework for EOS at this moment, the BlockSpace team will be using this framework to perform unit tests on smart contracts, given that the framework is not too quirky and unstable.

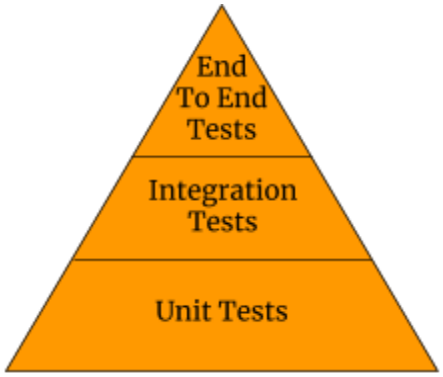


Figure 3: Three Layers of different testing levels

For the purpose of unit testing the Web Application, the project will use the Mocha[13] testing framework. This framework is very feature-rich and easy to use. Mocha works well with the most popular javascript assertion libraries. Mocha has a good testing coverage report feature as well as providing easy browser support.

To make writing and reading tests easier and clearer, some developers[16] prefer to use a framework on top of Mocha to add assertions with more syntactic sugar. For this the Chai[3] framework is a good choice. This framework integrates well with Mocha and adds extra functionality to the testing.

Integration Tests

In order to make sure that the interaction between the blockchain and the application is working correctly, it is also necessary to perform Integration tests. Integration tests from the Web Application can make calls to the blockchain which will have to be tested to make sure that these perform as expected.

The smart contracts can also perform integration tests with other blockchain aspects. Transactions can be simulated and their effects on smart contract tables can be tested using the unit testing framework used in EOS.

End To End Tests

To make sure that the User Interface (UI) is performing correctly, it is necessary to perform end-to-end testing. There are automated environments available to accomplish this. For the phaser library this could be done through a custom NightWatch extension phase-2-e[15]. The preference goes to automation. However, as it is not clear yet if automation will be feasible for the application, manual testing will also be an option. Manual testing would involve having a Test Document that will be created which provides a predetermined list of actions. At the end of each action or set of actions an assertion is made to check whether the desired result is achieved. At the end of each sprint/short development cycle the same manual tests should be performed. Each time the tests should pass the assertions.

Continuous Integration

In order to ensure that the code passes all tests as it progresses and that no failing code is uploaded to the repository, we will make use of Travis CI[19]. Although there are alternatives to Travis CI we decided to use this program, as all others are sidegrades, and we already have experience with Travis CI.

Conclusion

To study the best approach of building the BlockSpace blockchain game, we have looked and outlined the best development tools and testing practices.

To have an efficient workflow, the project will be developed using a lightweight SCRUM approach. This gives the best advantages for the project team, while also not requiring the team to spend too much time on documentation.

The project will use the EOS blockchain where the game logic will be stored on. The interaction with the game logic will be built as a website as this is the best option for using the current eosjs API.

References

1. airbnb/javascript. (2018). GitHub. Retrieved 4 May 2018, from: https://github.com/airbnb/javascript
2. BBC News. (2018). CryptoKitties cripple Ethereum blockchain. [online] Retrieved 9 May 2018, from: http://www.bbc.com/news/technology-42237162
3. Chaijs.com. (2018). Chai. [online] Retrieved May 4 2018, from: http://www.chaijs.com/
4. Denning, S. (2018). Agile: The World's Most Popular Innovation Engine, Retrieved 1 May 2018, from: https://www.forbes.com/sites/stevedenning/2015/07/23/the-worlds-most-popular-innovation-engine/
5. Dousos, F. (2018). What CryptoKitties can teach us about the success of blockchain - Dataconomy. [online] Dataconomy. Retrieved 2 May 2018, from: http://dataconomy.com/2018/01/cryptokitties-can-teach-us-success-blockchain/
6. Easycollect Services | Contactloos collecteren met pinbus | Collectebus 2.0. (2018). Easycollect Services. Retrieved 1 May 2018, from: https://easycollectservices.com/
7. Elahe, M., of Software Engineering, D., international University, D. and Mahmud, S. (2014). Efficiency of scrum the most widely adopted method for agile software development. [online] Academia.edu. Retrieved 1 May 2018, from: http://www.academia.edu/10923079/Efficiency_of_scrum_the_most_widely_adopted_method_for_agile_software_development
8. Eos.io. (2018). eosio | Blockchain software architecture. [online] Retrieved May 2 2018, from: https://eos.io/
9. EOSIO/eosjs. (2018). GitHub. Retrieved 4 May 2018, from: https://github.com/EOSIO/eosjs
10. Everts, V.(2018). Millions of transactions per second on the EOS.IO blockchain (INTERVIEW Ian Grigg) |. Vincenteverts.nl. Retrieved 2 May 2018, from: http://www.vincenteverts.nl/millions-of-transactions-per-second-on-the-eos-io-blockchain-interview-ian-grigg/
11. Google Developers. (2018). Introduction to Progressive Web App Architectures | Web | Google Developers. [online] Retrieved May 4 2018, from: https://developers.google.com/web/ilt/pwa/introduction-to-progressive-web-app-architectures

12. Lisk. (2018). Lisk. [online] Retrieved May 2 2018, from: https://lisk.io/
13. Mochajs.org. (2018). Mocha [online] Retrieved May 4 2018, from: https://mochajs.org/
14. Neo.org. (2018). NEO 智能经济. [online] Retrieved May 2 2018, from: https://neo.org/
15. npm. (2018). phase-2-e. [online] Retrieved May 4 2018, from: https://www.npmjs.com/package/phase-2-e
16. Patro, NC. (2018). JavaScript — Unit Testing using Mocha and Chai – codeburst. [online] codeburst. Retrieved May 9 2018, from: https://codeburst.io/javascript-unit-testing-using-mocha-and-chai-1d97d9f18e71
17. Promise. (2015). MDN-webdocumenten. Retrieved 4 May 2018, from https://developer.mozilla.org/nl/docs/Web/JavaScript/Reference/Global_Objects/Promise
18. Steem.io. (2018). Steem - blockchain-based social media platform. [online] Retrieved May 3 2018, from: https://steem.io/
19. Swende, M. (2018). Ethereum contract security. [online] Swende.se. Retrieved 3 May 2018, from: http://swende.se/blog/Breaking_the_house.html
20. Travis-ci.org. (2018). *Travis CI - Test and Deploy Your Code with Confidence*. [online] Retrieved May 4 2018, from: https://travis-ci.org
21. Trustnodes. (2018). Ethereum Tokens Now Account for 90% Market Share at $70 Billion. [online]Retrieved 2 May 2018, from: https://www.trustnodes.com/2018/01/13/ethereum-tokens-now-account-90-market-share-70-billion
22. Wright, A., & De Filippi, P. (2015). Decentralized blockchain technology and the rise of lex cryptographia.

END OF RESEARCH REPORT

D. Surveys

← Testing Phase 1

Testing Phase 2 →

Gameplay Experience Survey

This is an anonymous survey with the purpose of getting an idea of how the gameplay experience was while playing the early CryptoPIX prototype. The aim of the early prototype was to see if players would enjoy the idea of a daily pixel art drawing competition game concept.

* Required

How was your overall gameplay experience? *

	1	2	3	4	5	6	7	8	9	10	
Horrendous	○	○	○	○	○	○	○	○	○	○	Amazing

How many drawings did you make? *

- ○ 0-5
- ○ 5-10
- ○ 10-15
- ○ Other:

Did you like the addition of a daily competition? *

- ○ Yes
- ○ No
- ○ Don't Know

Do you think the daily competition added positively to the game experience *

- ○ Yes
- ○ No
- ○ Maybe

Was drawing pixel art an enjoyable experience as a game? *

- ○ Yes
- ○ No
- ○ Don't Know

Were the amount of colors good or would you like to have more/other colors *

- ○ The colors were sufficient for my drawing
- ○ It would be nice to have some more/other colors
- ○ The colors were not sufficient and more are definitely needed

Would you be interested in trying a more advanced form of this game? *

- ○ Yes
- ○ No
- ○ Maybe

Feedback or Comments

Your answer

SUBMIT

Never submit passwords through Google Forms.

CryptoPIX Gameplay Evaluation

This survey is aimed at getting feedback on the CryptoPIX gameplay experience.

* Required

Do you think that the game, including its relation to the game token, is difficult to understand? *

- ○ Yes
- ○ No
- ○ Maybe

Do you as a player, feel incentivized to win more game tokens? *

- ○ Yes
- ○ Somewhat
- ○ Not really
- ○ No

Do you believe that it is good, that your influence on the game is dependent on the amount of game tokens you have? *

- ○ Yes
- ○ No
- ○ Maybe
- ○ Other:

Do you feel as though your contributions to the canvas are rewarding? *

- ○ Yes
- ○ No
- ○ Other:

Does this application feel like a regular game? *

- ○ Yes
- ○ No
- ○ Maybe
- ○ Other:

Feedback / Comments

Your answer

SUBMIT

Never submit passwords through Google Forms.

E. Game Design Document

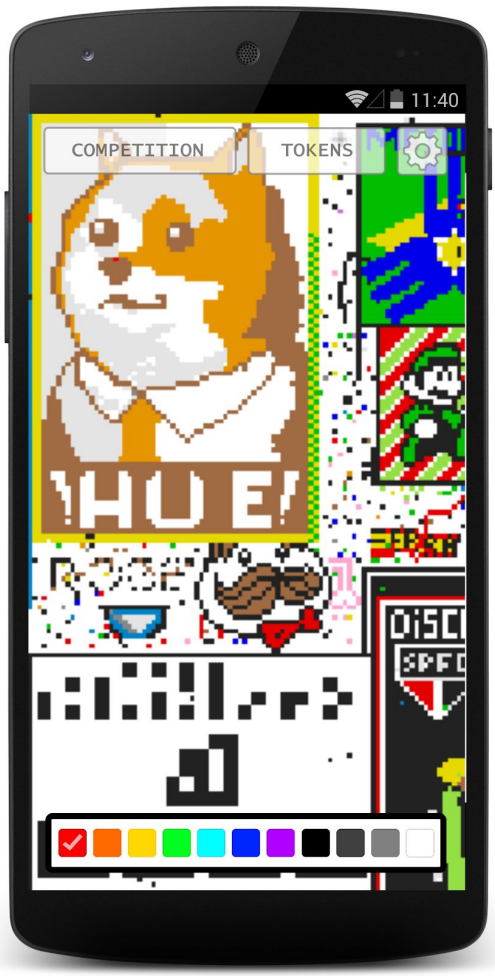
BLOCKSPACE

Game Design

Kees Fani, Pavel Hoogland, Manuel Ferreira

16-05-2018

Introduction



This document functions as the objective description of the game itself. Discussed in this document are the game mechanics, target audience, controls and visuals.

The reasoning for the game choices, can be found in the *Game Decision Reasoning* document. Details about its implementation can be found in the Research Report.

Game Overview

The game is a competitive cooperative online multiplayer drawing game. The game features a pixel canvas of $1000 \cdot 1000$ pixels. Every player gets to paint any arbitrary pixel on this canvas. The player can paint a pixel once every predetermined interval. The length of this interval is determined by the amount of *game tokens* the user has. The more tokens the user holds, the shorter the intervals will be. Users will be able to cooperate with a community (or create their own) to create whatever drawings they wish to make.

Every so often, an art competition will take place within the game. In this competition the players get to vote on the best drawing made during that time interval. The winners of this competition receive game tokens.

Target Audience

The target audience for this game is the general public, mainly casual gamers. It should feel like a regular mobile game, playable in any relaxed situation.

Technical Specifications

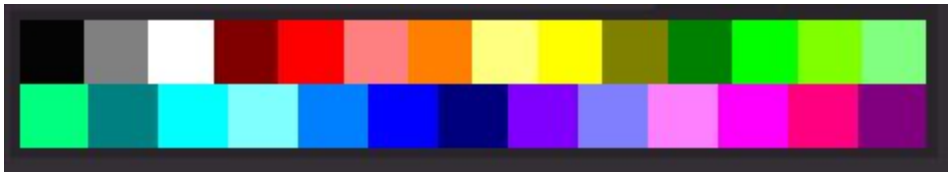
- The game is to be played on mobile devices and web browsers.
- Internet connection is a requirement.
- The game should be developed on the blockchain.

Main Mechanics

In order to describe the game more thoroughly, the main mechanics of the game will be laid out in this section.

Paint Pixels

When starting the game the players will see a grid game map for pixels. The players can select a color from a color palette, which they can use to paint a pixel on the game map. By clicking on the location on the map. This action is visible to all other players, and can be overwritten by anyone, given that they overwrite it with a different color. The palette of colors the player can choose from is displayed in the picture below. This palette is based on an old Amstrad CPC 8-bit computer.



After the player has painted a pixel on the map, they will have a cooldown period on the placing of a pixel. The cooldown will get shorter depending on the amount of tokens you hold as well as increasing during the day's game iteration (more on this in next sections) this will happen at a rate of: $\frac{1}{c \cdot (at^2+b) \cdot s}$ where c is the max amount of clicks per second in total we would want for the infrastructure, s is the players tokens staked (see tokens sections for more) percentage of all tokens staked, t is the time elapsed during the game iteration and a and b is determined by solving for our prefered end time interval at the end of the game iteration. (For further detail see *Game Decision Reasoning* document)

The maximum cooldown that you will have when you start the game with no tokens is a cooldown of 3 minutes.

Select drawings

Whenever a player is satisfied with the drawing they made, they can choose to submit them for a competition for others to vote on. This is done by going into 'selection mode' and selecting a start and end point on the canvas for the area of your submission. This submission can be of any size, so you can submit your little creations, as well as your big community effort drawings. When the deadline after 24 hours for that game iteration comes to an end, the drawing can no longer be modified, and the voting period can start for the canvas.

Vote on drawings

After the game iteration for the day has ended, a snapshot is taken of the canvas and the entire canvas is cleared for people to make new creations on. The goal of these short game iterations is to make the game have a fast paced competitive feel. After the canvas has frozen people are allowed to vote on the submissions for drawings from the snapshot. Players can switch to the submissions page where they will see all submissions of that iteration. Players can vote on any submission they like and can like as many submissions as they want to. The voting power of a player will increase linearly with the amount of tokens you At the end of this voting iteration, the drawings that have the most votes, will win, and receive tokens depending on their place and participation to the drawing.

Tokens

The game will be based on a token model. You will be able to earn, spend and stake/hold tokens. The player can earn tokens during the competition by having participated in the highest voted submissions. Giving out these rewards would mean that tokens would have to be created to pay out the rewards. As the game tokens would preferably have a max yearly inflation rate of 5%, the daily reward amount should be $\sqrt[365.25]{1.05} - 1 \approx 13500\ tokens$ With a total supply of 100.000.000 tokens. The player will earn rewards proportionally to their participation in creating the drawing.
After each pixel placement of a player they will get a cooldown until their next placement. If a player is in 'staking mode' , this cooldown will reduce at the rate mentioned earlier depending on the tokens you hold. This will encourage the player to hold many tokens so they have a short cooldown. Players can also choose to go into 'spending mode'. Now the player will be able to instantly place pixels without any cooldown, however they will spend 1 token for each pixel placed.

Game Play Flow

In this section we describe the various phases the game will have, in the order that players will most likely, participate in (although the contest phase could be done at any time).

First impression

When a player just starts playing the game, they will not have much drawing power, as such they will be easily overtaken by communities. Thus Solo play is not advised when playing this game as it will have a less enjoyable experience. Solo players could earn points by participating in the drawings from other communities and getting voted on, but this would not be as enjoyable as it turns the game into a ¨fill in the color¨ game instead of actually creating a drawing, and working on a goal.

Cooperation

Groups of players will have a linear increase of their drawing power in proportion to how many players there are in the group (group with 100 members has 100 times the drawing power of a single player). This will lead to these groups having the ability to create and protect their own drawings, allowing them to get voted and earn points.

Competitiveness

As the map has a limited space, there will be times when two or more communities will clash in order to complete their drawings, this will lead to a ¨war¨ in order to obtain supremacy over that area. The groups that work together the best and have the biggest community will eventually “win”.

Contest

The game will feature a drawing contest where people can submit drawings they made in the game. People can vote on the drawings they like the most. At the end of the contest period, those who helped on the winning drawing will get a token reward proportionally to their input on the drawing.

Controls

The game will be a simple Point-and-Click, consisting of 2 parts, painting and voting.

To paint a pixel on the map, players will have to select a color and the click on the map to paint. Everytime this is done, there will be a cooldown to stop players from spamming the entire map.

Voting would be done by having players submit an area of the map for voting, and then other players would be able to vote the selected area, giving points to everyone involved in the selected drawing.

Visualization

The game will have a minimalistic design, consisting of 3 main windows.

The first is the login screen, in which users can create their accounts/wallets to play the game.

The second is the main game window, in which the only thing the players see is the canvas, and a small box containing the color palette at the bottom of the screen. The player can zoom out on the canvas in order to see more of the drawing board, as well as being able to zoom in to more accurately place a pixel.

The third and final window, is the voting window in which the users will see a list of the submissions and vote on the ones they like.

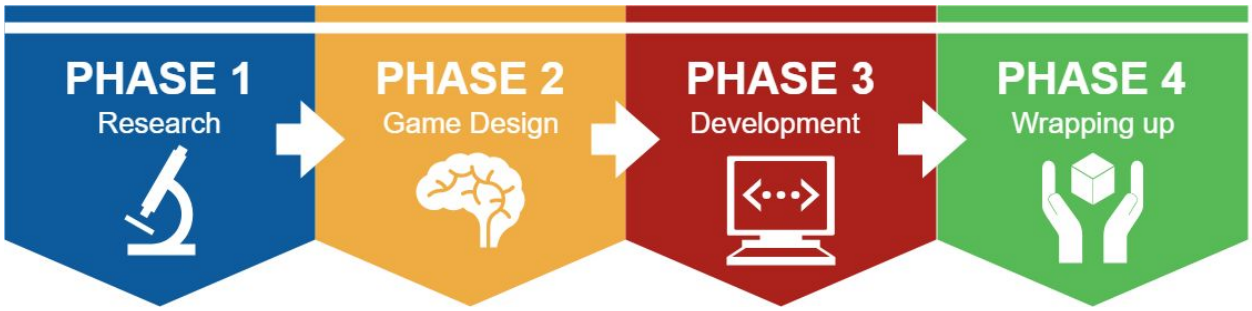
F. Roadmap

Roadmap Blockspace BEP

May, 2018

Introduction

This roadmap is used to gain a better understanding of the timeline of the Bachelor End Project (BEP). In this document we will describe what will be done in each phase of the BEP. First we will perform research to gain a better understanding of what, why and how the project will be completed. Then we will design and engineer the product itself. And finally we will present our BEP.



Phase 1 - Research - Week 1 & 2

The research process will consist of researching what the main goals of the game will be, and how those will be accomplished. Here we will also argue whether blockchain is a suitable technology for creating video games and which characteristics of a video game fit the best for the use of blockchain technology over traditional methods.

Researching How, Why and What to build - Week 1 & 2

In order to create the video game some research needs to be done beforehand. In this stage we will research how, why and what will be built, and on game design concepts in order to create a good final product.

Writing Research document - Week 2

At the end of week 2 we shall deliver a research document containing all the collected information relevant to the project. This document should have enough information to provide a solid foundation for the creation of the game.

Phase 2 - Game Design - Week 3

In this phase, the game design creative process will take place. The client and the development team will put their minds together to come up with a good and fun game concept. This creative process will be documented in the Game Design Document. This includes the reasoning for the choices as well as a detailed description of the game concept itself.

Phase 3 - Development - Week 4 to 9

The development of the project will primarily be divided in two sections. Firstly the game application logic, which will be in the blockchain. Secondly the interface the end users will interact with, which will be a mobile (web-)application.

In week 6 (1st of june), the first Software Improvement Group (SIG) submission will take place. This submission should contain a working solution build on top of the EOS blockchain platform as well as having the mobile application solution.

Blockchain development - Week 4 to 7

Creating the blockchain solution for the game will be of utmost importance. The blockchain is where the token currency and game application logic will reside.
For this reason, blockchain development will start first and have greatest priority.

In the 7th week of the blockchain development, the feedback of SIG's analysis will be implemented.

Mobile App/Web development - Week 6 to 9

End users will need an interface to interact with the blockchain game application. In order to make this as accessible as possible, a mobile (web-)application will be built. The development of this application will have a two week overlap with blockchain development.

Finally the software will be submitted to SIG at the end of week 9 (22th of June).

Writing Thesis - Week 5 to 9

In order to formalize our work, we will write a thesis consisting of what we did and how we did it. We will also describe the reasoning behind our choices.

Since we want to describe our work as well as possible, we will write the thesis in a timespan of five weeks. Writing this thesis will be a continuous process starting early in the development of the software and ending when we finish development.

Phase 4 - Wrapping up - Week 10 & 11

In this final phase the main objective is to deliver the final product and prepare for the final presentation of the BEP. As such it will consist mainly of finishing touches on the game and creation of the presentation.

Finishing up - Week 10

Here we will implement the feedback from the SIG analysis and provide the finishing touches for the project. When this is done the project will be given to the client.

Presentation - Week 10 & 11

Finally we will present our work during a 30 minute presentation period. This will happen in the final week. We will prepare the presentation one week in advance.

G. Infosheet