“ORB”

DECENTRALIZED FUNCTION AS A SERVICE (DFAAS)

Rewarding System

Project ID – 18-070

Preliminary Progress Review

A.T.Nimansa

Bachelor of Science Special (Honors) Degree in Information Technology

Department of Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

March 2018

Contents

Table of Figures

1.0 Introduction

Decentralizing server space to host a function makes services provided more reliable. Decentralized Functions as a service or in one term, “Orb “grant the user the privilege of enjoying a reliable secure and cost-effective service. The purpose of “Rewarding system” in orb is simply to provide it’s users who act as function seeders, owners and requesters a payment model which acts in a logical manner to provide fair rewards to all.

1.1 Purpose

Purpose of this document is to justify the importance of the Rewarding system to our research and how it actually works. By reading this one can understand the logic behind the payment model and difference it has from typical payment methods. Through this document the similar systems, our technical objectives and detailed design of the rewarding system in Orb will be discussed. The important tasks like resource management and sample user interfaces are also included in here.

1.2 Scope

Scope of the Rewarding system component is a collection of several activities. Payment Model based on requests served by a node. The functionalities our rewarding system should possesses includes,

I. Finding out the possibility of paying the nodes which hosts functions based on served requests

The function owner initially hosts his own function and if some user is interested in hosting it, he can seed it to be used by requesters. When requester, request for a specific function the function owner and the seeder both gets rewarded as per the requests for a pre-defined ratio

II. Finding out a mechanism to validate the requests served by a mechanism between the origin of the request and the requests served

Validating the requests mainly focuses on blocking, function owner or seeder to send requests to it's own function and corrupt the rewarding system. For the above-mentioned function finding out a secure mechanism to prevent DoS attacks which lets a node earns money by sending pings should be done

III. Creating a portal to deploy smart contracts which works as price models

A smart contract is a computer protocol intended to digitally facilitate, verify, or enforce the negotiation or performance of a contract. Smart contracts allow the performance of credible transactions without third parties. These transactions are trackable and irreversible. This is needed when we need to distribute the rewards to the users. We can implement the logic of price models via the smart contract and distribute the rewards. But in our research, it is not possible to create a exact smart contract with Ethereum as the paying unit since we do not own any Ethereum crypto currency. For that we will create a program which acts as a stimulator.

IV. Finding a method to determine the value of a smart contract.

The nodes might host different kinds of functions with different complexities. It is not fair to treat each and every function the same and reward equally. There should be a method to estimate the complexity of the function and calculate the value of the function accordingly. There are two main sections that we should find a way to calculate the value of

- The value to pay for the function owner
- The value to pay for the function seeders
- The value per request for requestor

1.3 Overview

1.3.1 Main Goals

I. Provide a payment model

- Pay the function owner
- Pay function seeders
- Charge from function requesters

II. Calculate the complexity of the functions

Calculating the complexity of the functions is important to give a value to each function. There can be many methods to calculate it such as

- Function execution time
- Lines of code
- Size of the function

III. Calculate the market value of the function taking the complexity of the function as the base

Since the functions are distributed we should use one of the above-mentioned methods which will give the approximate comparisons of the complexities and let us use the value generated to get a price.

IV. Automatically distribute the rewards

There should be a way to identify the connected nodes, and filter the seeders, owners separately. When the function is requested by another node the number of requests for the function should be counted and then the rewards should be distributed. For this we should create a program which will act as an intermediary contract to autonomously charge the prices and grant the rewards

2.0 Statement of work

2.1 Literature survey

There are other distributed networks built for many other purposes but the specialty in Orb is it provides Function as a Service.

These are some platforms, browsers, protocols build on top of decentralized networks

I. Steemit

Steemit is a blogging and social networking website that uses its Steem blockchain-based rewards platform for publishers. It gives steem as the reward for the users. But in this case there is no such thing as hosting a function but it's rewarding system has similarities with Orb.

II. Storj

It is a platform, cryptocurrency, and suite of decentralized applications that allows user to store data in a secure and decentralized manner. Storj can be faster, cheaper, and more secure than traditional cloud storage platforms. Faster because multiple machines are serving user his file simultaneously, cheaper because user is renting people's spare hard-drive space instead of paying for a purpose-built data center, and more secure because user's file is both encrypted and shredded.

III. IPFS

InterPlanetary File System is a protocol and network designed to create a content-addressable, peer-to-peer method of storing and sharing hypermedia in a distributed file system. This is more of a protocol which has many differences than Orb, since Orb is providing functions as a service

IV. Beaker browser

Beaker rethinks the Web as a peer-to-peer network, where users own their data and run applications independently. Beaker is the world's first Web browser with builtin tools for hosting content directly from user’s computer.

2.2 Identification and significance of the Problem

All the above-mentioned systems are Static content providers in the world. But with Orb user can use dynamic backend services which act as FaaS. If we take it in rewarding system aspect only, the more similar system is Steamit. It has it’s own kind of rewards called steem and when users share, comment on their social media platform they get rewarded.

Our system is not having anything related social media but the rewarding system is somewhat similar to Steemit. Though steemit uses it for marketing purpose in Orb it acts as the decentralized payment model when it provides Functions as a service

In storj also they act as a system which provides server space but the main dissimilarity is we use it to provide it’s uses to host functions and earn out of it. Storj is the data layer for the Internet. Decentralized cloud storage is a new paradigm that removes intermediaries. Storj is open source, distributed, encrypted, and blazing fast object storage. It is more of a storage space and does not have any feature to host a function and give another user an end point to use it like in Orb.

2.3 Technical objectives

The software requirements it needs the server software to be installed on a server with following requirements.

1. Dual Core CPU
2. 4GB Ram
3. Public IP Address

Client software is available for any platform. Client software is made using electron framework. It's supporting cross platform.

This system needs a computer with 4Gb ram and i3 processor as minimum requirements. Consumer can use a desktop or laptop as a device.

User needs to download client software and install it. Then client software will generate a key pair and a unique id for each client. User needs to do this only one time. Generated id is unique for a client and it's a must to keep it safe. If the user is installing the client software for the second time, no need of creating an id again. User can upload the key pair and get sign in. So key pair act as an entry key for the whole system.

Client software generates the key using several parameters (e.g.: timestamps, email). One single node (consumer device) is recognized as user at a time. Consumer device (node) is simply a Desktop PC, laptop. Later we can extend this to IoT and mobile devices, since infrastructure is common for all. For an example if we consider a PC it need to have normal specifications like i3 or i5 processor, 4GB ram etc.

2.4 Detail Design

2.4.1 What is the rewarding system

Rewarding system is the payment model in Orb. Users won't deploy their functions or seed someone else function without some gain. This rewarding system is there to encourage the user earn out of providing a useful function and pay as they go.

2.4.2 How Rewarding system works

To get an idea how our rewarding system actually works, we should try this out with an example

Example –

Initially all the users get 50 units (currency)
When deploying cost will be 10 units
When requesting, requester's units will be reducing by one, and deployer's units will be increased by one

If we have a table for each phase

Phase	Deployer	Requester
Registration	+50	+50
Deployment	+50 -10 = +40	+50
Request	+40 +1 = +41	+50 -1 = +49

Same scenario explained using a diagram

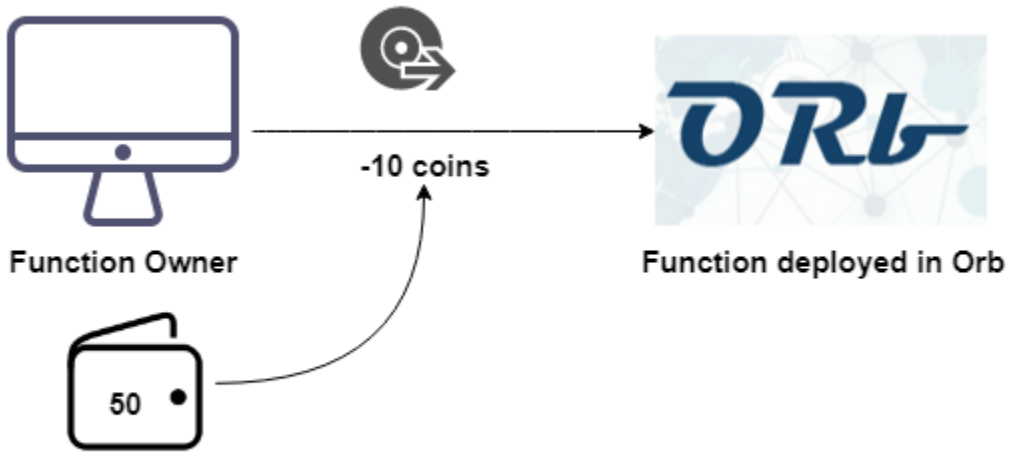

Figure 1- Deploying a function in Orb

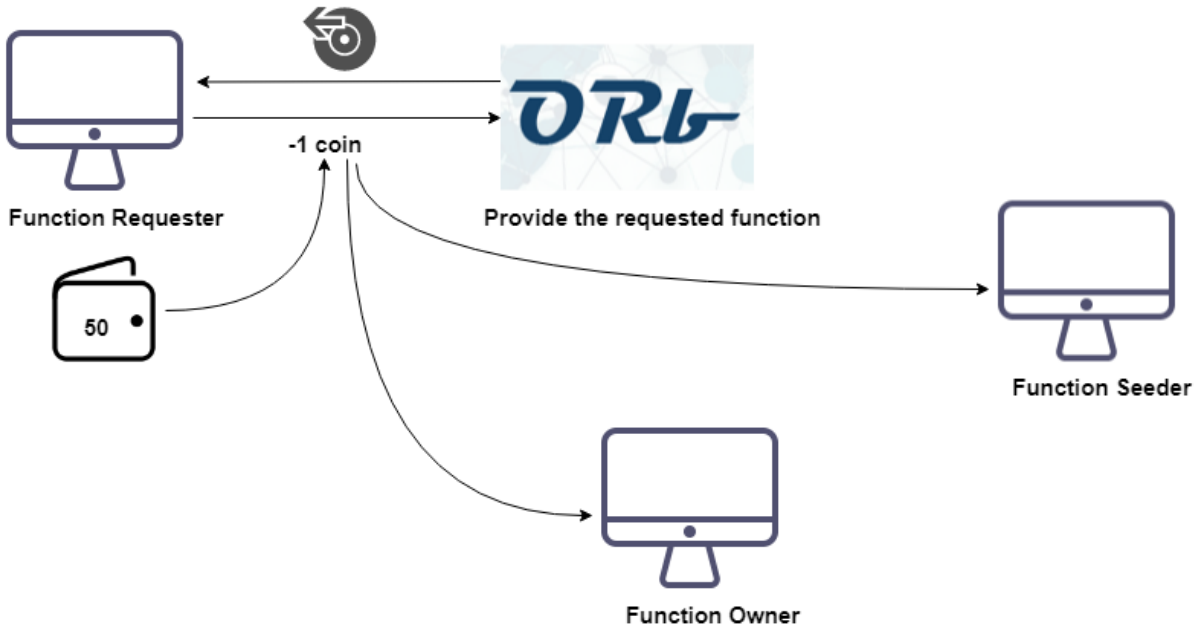

Figure 2 - Getting rewards as per request

If we take this in different use cases it will give a better understanding

Function owner

I. Get registered to Orb
II. Get a reward of 50 coins initially
III. Create a function
IV. Pay the deployment cost and upload the function to the “Orb”
V. Get 10^6 X 2 amount of free requests to his own function
VI. Get paid if the function get requested by other users for 10^6 times (in 2:8 proportion with seeders)

Function seeder

I. Get registered to Orb
II. Get a reward of 50 coins initially
III. Select a function to seed
IV. Check for the resources needed and if they match download and seed it
V. Get rewarded when 10^6 requests from requesters are received
VI. Get rewarded from the deployment cost allocation

Function requester

I. Get registered to Orb
II. Get a reward of 50 coins initially
III. Request for functions
IV. Pay when they reach 10^6 requests

2.5 Anticipated benefits

When Orb is taken as a system there are many benefits that a user can gain. But in this context only the benefits of having this rewarding system is explained

I. Can earn for what you do

If a person implement some function that is useful to everyone but not having enough resources to host it keep it available he should use a services provider like AWS. The best example for such type of a service is AWS Lambda

AWS Lambda lets the user run code without provisioning or managing servers. User pay only for the compute time he consume. With Lambda, user can run code for virtually any type of application or backend service - all with zero administration. Just upload his code and Lambda takes care of everything required to run and scale your code with high availability. Although it has many advantages and high availability, the cost will be high when the scaling.

In Orb scaling is allowed with lesser cost and also allows the user to earn when other peers use their function.

II. Can use the resources in an efficient manner

Almost everyone have a computer or a smart device. Most of them are for personal use and there are time periods which the computer is running but without proper use. Orb let the user to use these resources efficiently and earn out of it

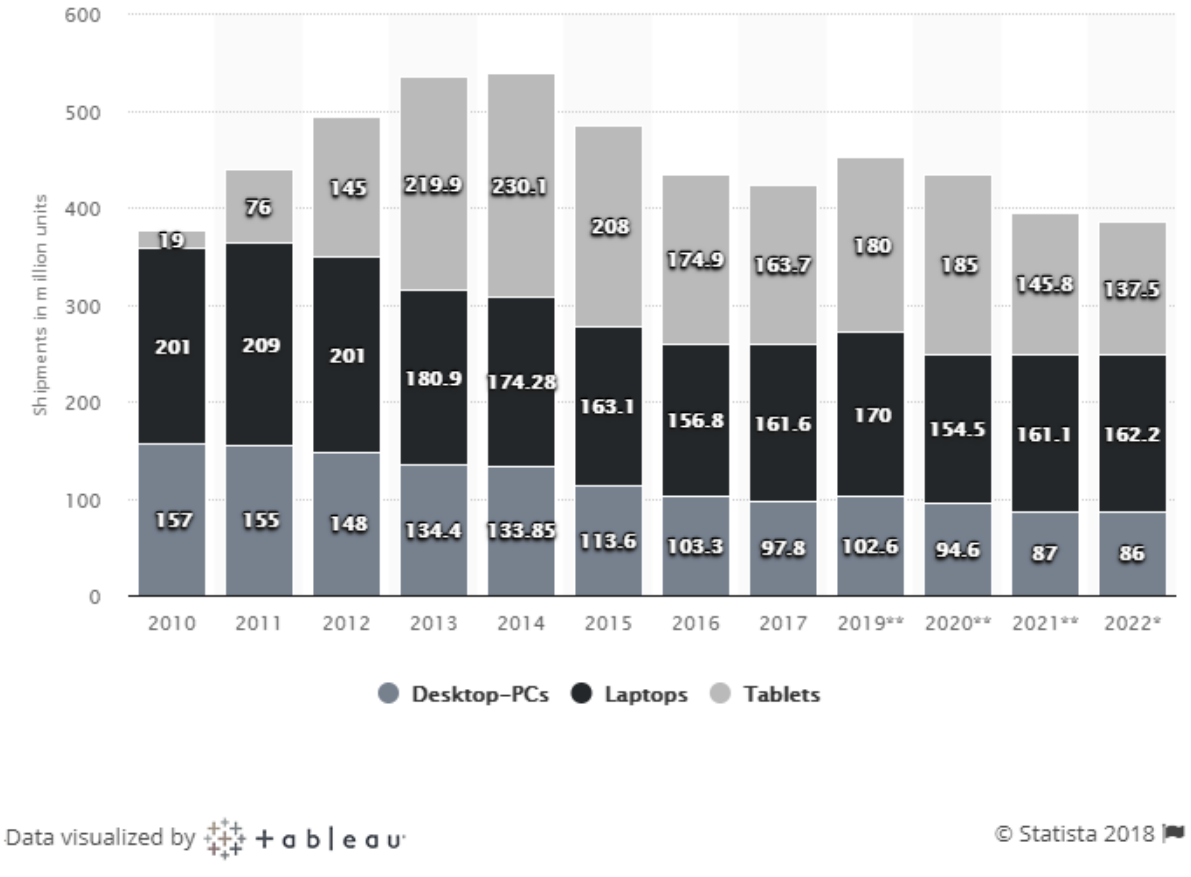

Figure 3 - Shipments of PCs, Laptops and tablets

As per statista.com the above bar graph shows the number of devices with it's respective amounts that is shipped worldwide during 2010- 2022. The resources that are available will be huge and due to the attractive rewarding system Orb has there is a high tendency to grow in the market

III. Can reuse what you earned to get services from other nodes

For the coins you earn and or the coins you spend there should be a tracker. We introduce a e-Wallet where everything the user spent and earn to be tracked. When a function owner earns as per the requests he get he can either use it to get another type of service, or else convert the coins to a normal currency

Though we only implement a rewarding system using a coin we created to stimulate the same procedure as a smart contract, in real scenario, the crypto currency Ethereum can be used. If the rewards users' gets are eth they can gain high profits with the value fluctuation of it

3.0 Project plan

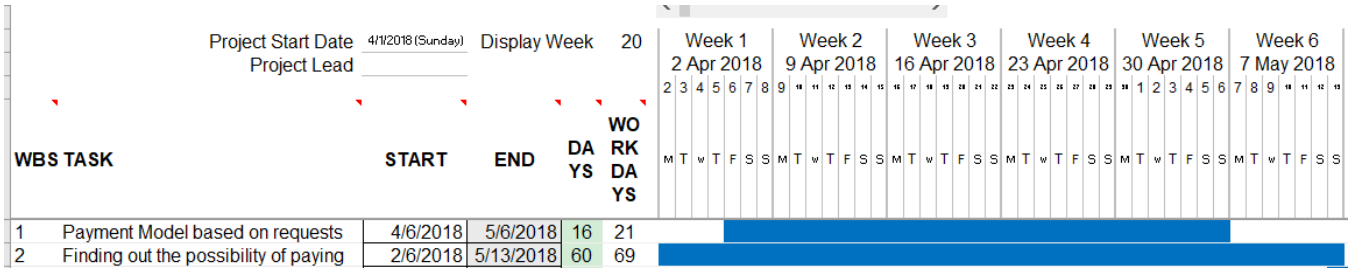

Figure 4 - Gantt Chart (Part 1)

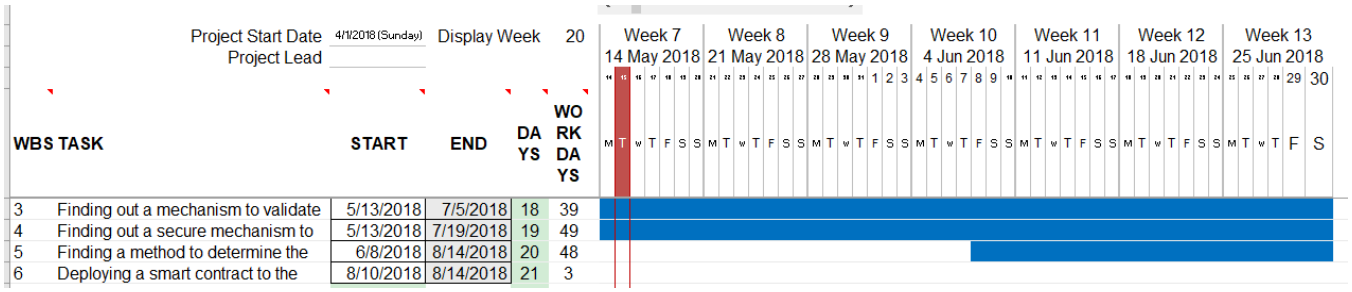

Figure 5- Gantt Chart (Part 2)

4.0 Research constraints

The main research constrain is the rewards division method

The logical and effective method of implementing such system to distribute rewards is having a smart contract and using Ethereum as the transaction unit. As explained in previous sections smart contracts are programs that execute exactly as they are set up to by their creators (autonomously) and acts as a normal contract. Ethereum is a platform that's built specifically for creating smart contracts. But these new tools aren't intended to be used in isolation. It is believed that they can also form the building blocks for 'decentralized applications'

Smart contracts can:

I. Function as 'multi-signature' accounts, so that funds are spent only when a required percentage of people agree
II. Manage agreements between users, say, if one buys insurance from the other
III. Provide utility to other contracts (similar to how a software library works)
IV. Store information about an application, such as domain registration information or membership records.
V. Example - Ethereum user can send 10 ether to a friend on a certain date using a smart contract

The limitation we have is we don’t own Ethereum and smart contract implementation is not possible without it. Instead of a smart contract our idea is to give the same kind of stimulation by creating our own transaction unit.

5.0 Specified deliverables

The deliverable in this section is a software which act as a wallet to the user. Using this wallet the user will be able to track all his activities, get total rewards, get total expenses and let the user have a great user experience with high security and low risk.

5.1 Orb Wallet

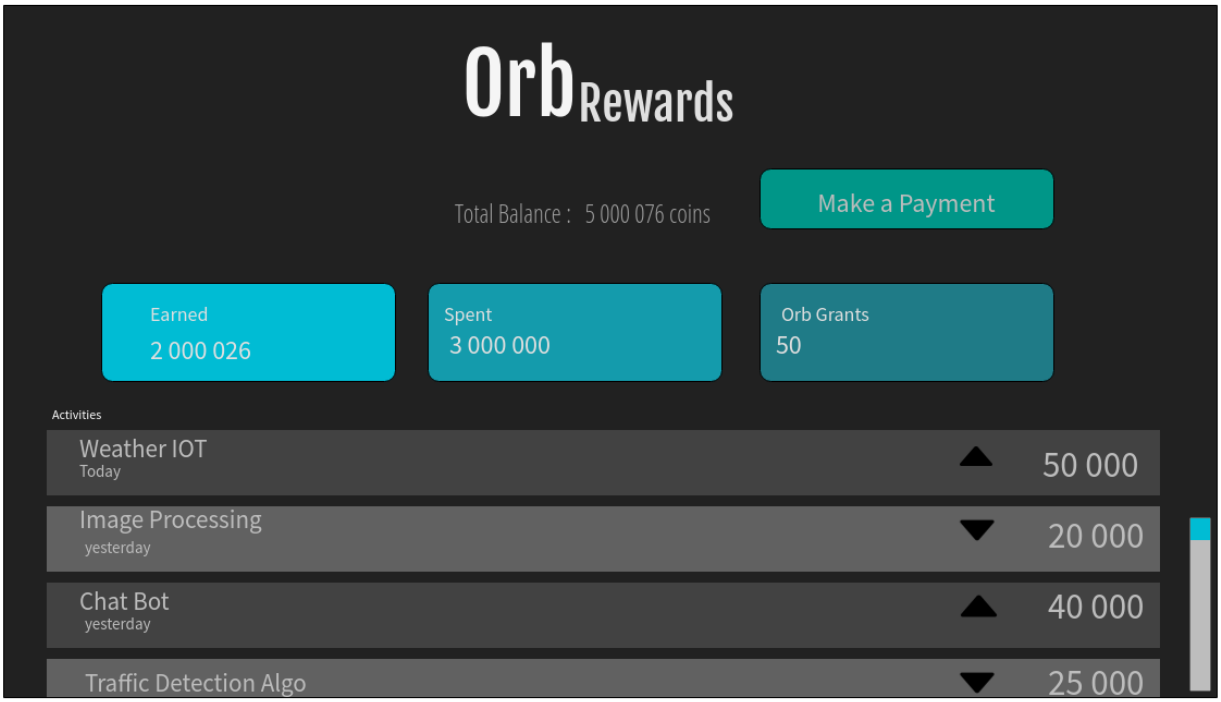

Figure 6- Orb Wallet Home page

Orb wallet is an application that will,

i. Let the user get registered and login to make transactions
ii. Show the list of transactions occurred in the past
iii. Show the total reward earnings and deductions
iv. Calculate the deployment cost for the functions that are to be uploaded
v. Notify the seeders and function owners when they get paid

The Orb wallet will be able to get installed once Orb client software is installed

6.0 Supporting Information

6.1 References

[1]D. Etherington, "Amazon AWS S3 outage is breaking things for a lot of websites and apps", *TechCrunch*, 2018. [Online]. Available: https://techcrunch.com/2017/02/28/amazon-aws-s3-outage-is-breaking-things-for-a-lot-of-websites-and-apps/. [Accessed: 16- Jan- 2018].

[2]M. Roberts, "Serverless Architectures", *martinfowler.com*, 2018. [Online]. Available: https://martinfowler.com/articles/serverless.html#unpacking-faas. [Accessed: 14- Jan- 2018].

[3]"What is Ether", *Ethereum.org*, 2018. [Online]. Available: https://www.ethereum.org/ether. [Accessed: 18- Jan- 2018].

[4]"How Do Ethereum Smart Contracts Work? - CoinDesk", *CoinDesk*, 2018. [Online]. Available: https://www.coindesk.com/information/ethereum-smart-contracts-work/. [Accessed: 14- Jan- 2018].

[5]"ConsenSys – Medium", *Medium.com*, 2018. [Online]. Available: https://medium.com/@ConsenSys. [Accessed: 21- Jan- 2018].

[6]"What Is AWS Lambda? - AWS Lambda", *Docs.aws.amazon.com*, 2018. [Online]. Available: https://docs.aws.amazon.com/lambda/latest/dg/welcome.html. [Accessed: 22- Jan- 2018].

[7] “Steemit,” *Steemit*. [Online]. Available: https://steemit.com/. [Accessed: 24-Mar-2018].

[9]"AWS Rates Highest on Cloud Reliability", *EnterpriseTech*, 2018. [Online]. Available: https://www.enterprisetech.com/2015/01/06/aws-rates-highest-cloud-reliability/. [Accessed: 14- Mar- 2018].

[10]"The AWS Outage: The Problem with Internet Centralization | Mondo", *Mondo*, 2018. [Online]. Available: https://www.mondo.com/aws-outage-internet-centralization-problem/. [Accessed: 14- Jan- 2018].

[11]"Single Point of Failure – What is it? Why it Matters… :", *Renovodata.com*, 2018. [Online]. Available: http://www.renovodata.com/blog/2015/06/10/single-point-of-failure. [Accessed: 14- Jan- 2018].

[12]"www.ey.com", *Ey.com*, 2018. [Online]. Available: http://www.ey.com/Publication/vwLUAssets/EY-implementing-blockchains-and-distributed-infrastructure/$FILE/EY-implementing-blockchains-and-distributed-infrastructure.pdf. [Accessed: 14- Jan- 2018].

[13]P. Labs, "IPFS is the Distributed Web", *IPFS*, 2018. [Online]. Available: https://ipfs.io/. [Accessed: 14- Jan- 2018].

[14]"Storj - Decentralized Cloud Storage", *Storj - Decentralized Cloud Storage*, 2018. [Online]. Available: https://storj.io. [Accessed: 14- Jan- 2018].

[15]*Google Cloud Platform Documentation | Documentation | Google Cloud*. [Online]. Available: https://cloud.google.com/docs/. [Accessed: 02-Feb-2018].

[16]"Kubernetes Documentation," *Kubernetes*. [Online]. Available: https://kubernetes.io/docs/home/?path=users&persona=app-developer&level=foundational. [Accessed: 02-Feb-2018].

[17]"Decentralized Internet on Blockchain – Hacker Noon," *Hacker Noon*, 03-Oct-2017. [Online]. Available: https://hackernoon.com/decentralized-internet-on-blockchain-6b78684358a. [Accessed: 02-Feb-2018].

[18]*Blockchain Based Distributed Control System for Edge Computing - IEEE Conference Publication*. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/7968630. [Accessed: 02-Apr-2018].

[19]"Decentralizing Your Microservices Organization," *The New Stack*, 31-Oct-2017. [Online]. Available: https://thenewstack.io/decentralizing-microservices-organization/. [Accessed: 02-Apr-2018].

[20]"Building Your First Network," *Building Your First Network - hyperledger-fabricdocs master documentation*. [Online]. Available: http://hyperledger-fabric.readthedocs.io/en/release-1.0/build_network.html. [Accessed: 02-Apr-2018].

[21]M. Meister, “leveraging Kubernetes to run a private production ready Ethereum network,” *Medium*, 26-Nov-2017. [Online]. Available: https://medium.com/@cryptoctl/leveraging-kubernetes-to-run-a-private-production-ready-ethereum-network-b6f9b49098df. [Accessed: 02-Apr-2018].

[22]*A Decentralized Service Discovery Approach on Peer-to-Peer Networks - IEEE Journals & Magazine*. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/5928313/. [Accessed: 02-Apr-2018].

[23]“Taming WebRTC with PeerJS: Making a Simple P2P Web Game,” *Toptal Engineering Blog*. [Online]. Available: https://www.toptal.com/webrtc/taming-webrtc-with-peerjs. [Accessed: 02-Apr-2018].

[24] “Decentralized payments for environmental services: The cases of Pimampiro and PROFAFOR in Ecuador,” Ecological Economics, 31-Dec-2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921800907005320. [Accessed: 05-Jan-2018].

[25] A. Crespo and H. Garcia-Molina, “Semantic Overlay Networks for P2P Systems,” *SpringerLink*, 19-Jul-2004. [Online]. Available: https://link.springer.com/chapter/10.1007/11574781_1. [Accessed: 02-Mar-2018].

[26] “Rethink the Web browser,” Beaker | Peer-to-peer Web browser. No blockchain required. [Online]. Available: https://beakerbrowser.com/. [Accessed: 05-Feb-2018].

[27]"Statista - The Statistics Portal for Market Data, Market Research and Market Studies", Statista.com, 2018. [Online]. Available: https://www.statista.com/. [Accessed: 15- May- 2018].

[28]"How Do Ethereum Smart Contracts Work? - CoinDesk", CoinDesk, 2018. [Online]. Available: https://www.coindesk.com/information/ethereum-smart-contracts-work/. [Accessed: 15- May- 2018].

6.2 Appendices

6.2.1 Wire Frames

6.2.1.1 Home Page

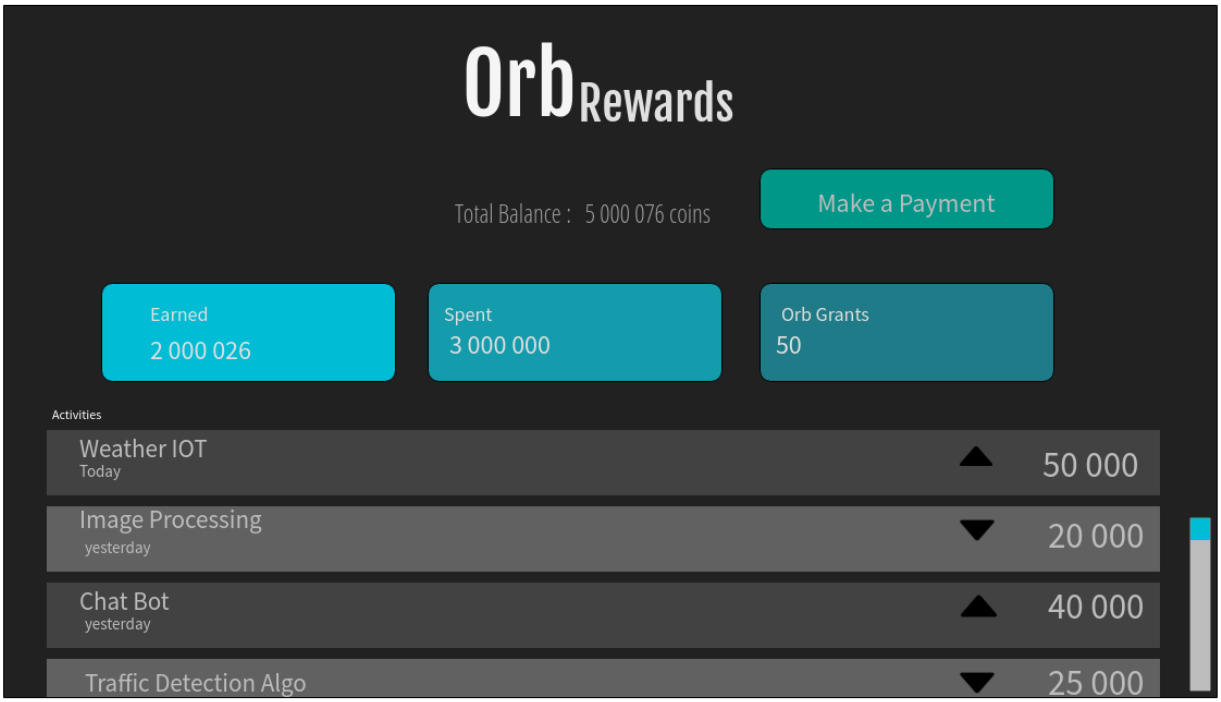

Figure 7- Home Page

Home page of Orb Rewarding system will have a similar look as above. The total earned coins spent and free grants by Orb will be displayed to the user at a glance. All transactions will be listed as activities and user can simply make a payment by clicking the “make a payment” button.

6.2.1.2 Paying to deploy a function

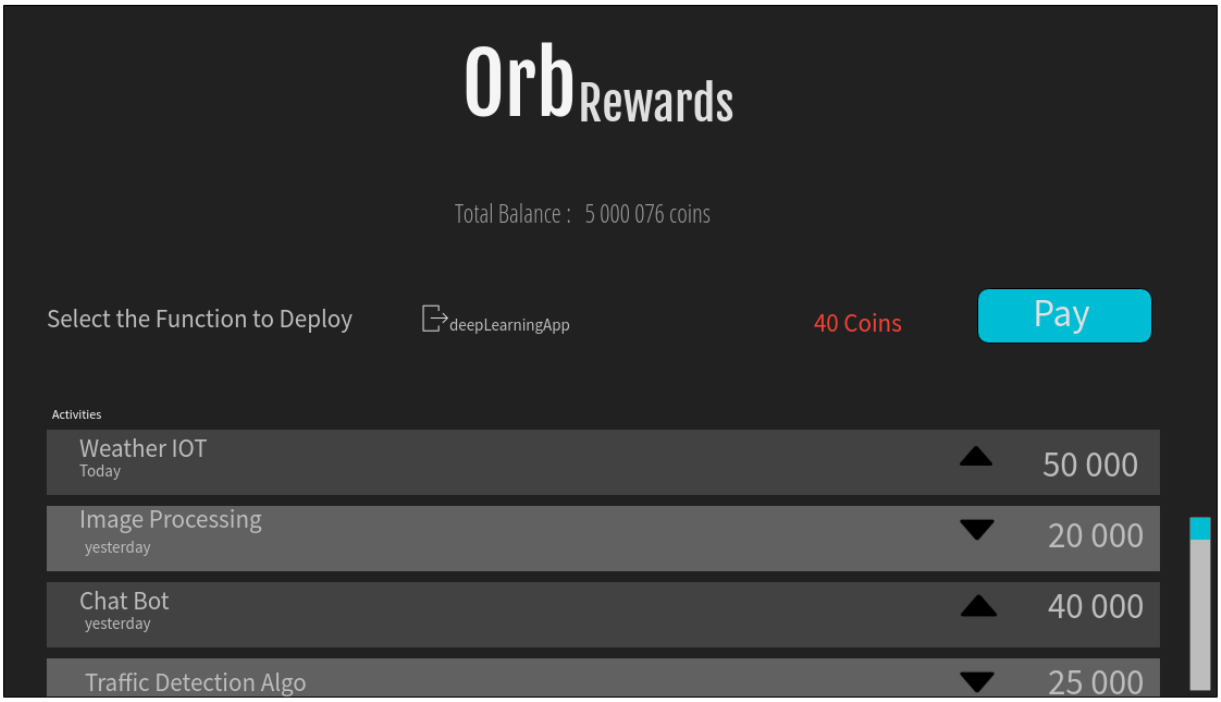

Figure 8- Paying to deploy a function

A user should be able to pay for orb when they are uploading a function to the network. The amount will be calculated and displayed, allowing user to pay just by one click

6.2.1.3 Orb Notification

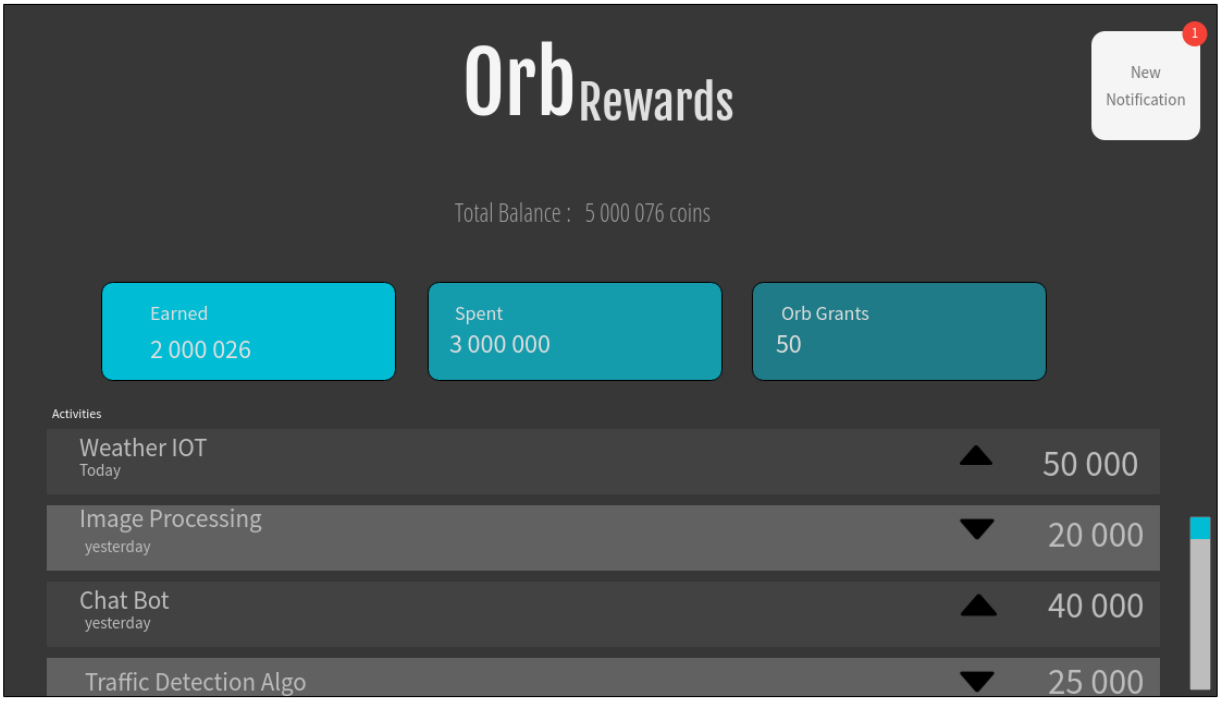

Figure 9 -Orb Notification

New notification will be displayed when a user gets some reward or deduction occurs. User can simply view the notification by clicking it or when the time is out those will be displayed under activity list

6.2.1.4 Orb notification display

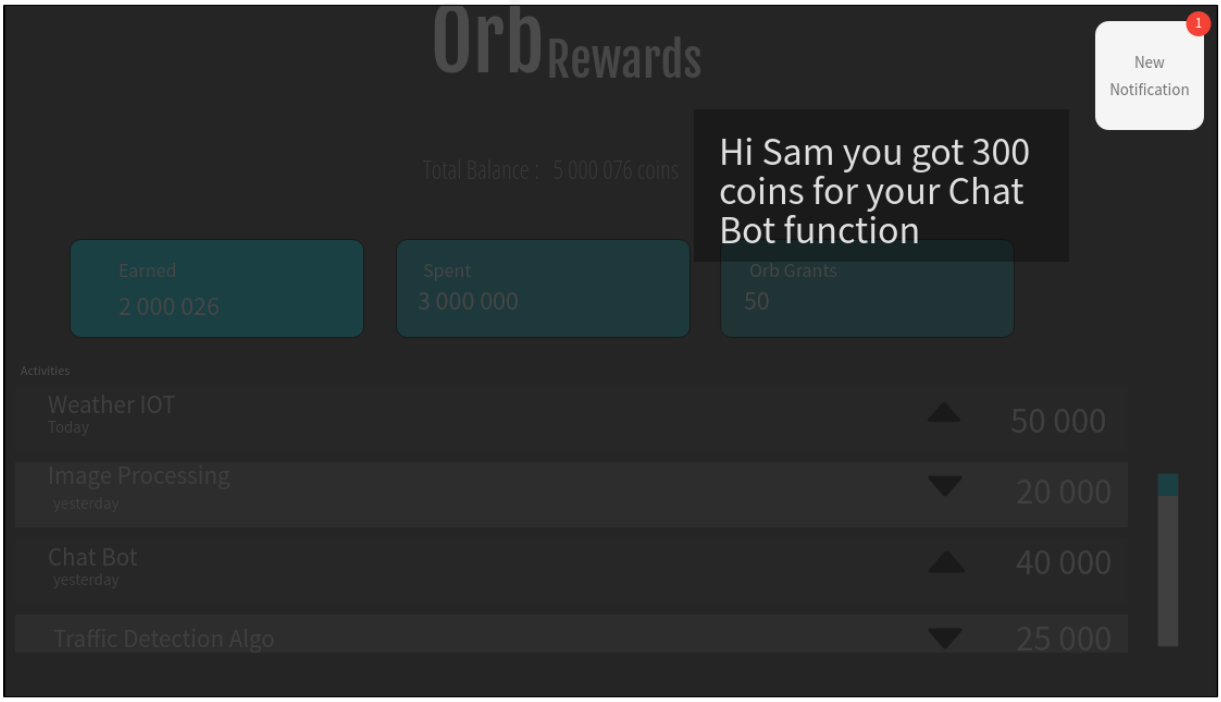

Figure 10- Orb Notification display

The notification can be viewed to the user as mentioned earlier. And once user click on the background it will display the normal home view