DAMP – Digital Asset Management Platform

by

Austin Jaeger, Daniel Richter, & Michael Wood

Submitted to
the Faculty of the School of Information Technology
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Technology

Austin Jaeger	04/13/2020
Austin Jaeger	Date
Daniel Richter	04/13/2020
Daniel Richter	Date
Michael Wood	04/13/2020
Michael Wood	Date
Bo Vykhovanyuk	04/13/2020
Bo Vykhovanyuk	Date

University of Cincinnati
College of
Education, Criminal Justice, and Human Services

April 2020

TABLE OF CONTENTS

List of Illustrations

TABLES

FIGURES

ABSTRACT

In today's market, large game distribution platforms take a sizable cut of developers' game sales, which significantly impacts indie developers' ability to grow. In the recent GDC State of the industry that was held in March of this year, it was polled that "just 6% of nearly 4,000 respondents believe that Steam justifies the 30% cut it takes from developers"(Sinclair 2019). Our team will create a blockchain network based on Ethereum that will function as the ledger and transaction engine for our digital game distribution platform. We are developing this platform to create a marketplace focused on allowing indie game developers to earn a larger cut of their sales. In addition, our platform will provide infrastructure by crowd-sourcing hardware from nodes in our network by incentivizing users to host content and provide downloads in return for a small gratuity. Through a combination of Hyperledger frameworks and IBM blockchain technology, we will be able to minimize the middleman and shrink the necessary infrastructure to a fraction of traditional methods.

1. Project Introduction

1.1 Introduction

Blockchain as a technology has existed for over a decade now. Invented in 2008 by someone under the alias of Satoshi Nakamoto, it was originally used as a public transaction database for cryptocurrency. Today, blockchain has exploded in popularity and is used in private and public applications alike, from cryptocurrency to supply chain to video games. Our goal is to apply blockchain in an area that could be significantly changed by this technology.

1.2 Problem

In 2018, Steam's total account base was recorded to be 125 million. On average, 90 million users are active monthly and 45 million daily. Users are doing everything from shopping, gaming and socializing in Steam's vast application. Like many companies today, Steam and other gaming platforms require a large amount of infrastructure to run and therefore must cover those expenses somehow. These platforms cover operating costs by taking a percentage of a developer's game sales made via the platform.

For example, Steam takes a significant cut on all games sales ranging between 20% to 25% based on the game market performance (Vincent 2018). This puts a strain on small developers that are trying to break into the gaming market. To date there are only a few solutions focused on helping these small developers get up on their feet such as Itch.io and

Steam Greenlight. Yet, none of them have gained enough traction to adequately support the small developer community.

1.3 Solution

With the advent of blockchain technology, countless industries have seen a revolution in how data, transactions, and ledgers can be handled. Our solution, the Digital Asset Management Platform, will address the oversight and lack of support for small developers that game distribution giants like Steam and Origin have failed to acknowledge. Our platform seeks to provide these small developers with an open marketplace to sell, advertise, and distribute their games; and at a fraction of the cost with which these industry giants do.

With the inherent nature of blockchain, our product not only has built in additional layers of security but the capacity to provide this platform at significantly lower operating costs. DAMP will benefit both gamers and developers by lowering the impact of the middleman during transactions. This will give back more revenue to the developers, and by extension, create a free marketplace which will result in lower, more competitive prices to customers when compared to Steam and Origin.

1.4 Project Description

Our team will create a blockchain network based on Ethereum that will function as the ledger and transaction engine for our digital distribution platform. To accomplish this, we will be using a combination of Hyperledger frameworks and IBM blockchain technology. We will also create a web application to interface with this network, allowing consumers and developers to browse and sell games together by using smart contracts, a functionality that is built-in to Ethereum. By combining the frontend interface with the blockchain ledger on the backend, we will be able to create a fully immersive game marketplace with none of the infrastructure of a traditional digital distribution platform.

1.5 Overview

The remainder of this final report outlines in detail how the project was completed. The report includes the following sections: design objectives, methodology, budget, timeline, problems encountered, and future recommendations.

1.6 User Profile

We have determined that DAMP with have two kinds of users interacting with the front end: Customers and Admin/Devs for the blockchain. These customers will be individuals interested in purchasing video games using Ethereum cryptocurrency.

1.6.1 Project Title

DAMP- Digital Asset Management Platform

1.6.2 Potential Users

- Admins
- Developers
- Users

1.6.3 Software and Interface Experience

Our user clientele will be oriented towards regular gamers and how they interact with purchasing current games through existing platforms. The user will require basic knowledge of purchasing a game and how to apply a game key to their account. Having a basic knowledge of cryptocurrency would not be required but could allow for a user to have a better understanding of how the platform works.

1.6.4 Experience with Similar Applications

- Steam
- Origin
- Epic games
- Blizzard launcher

1.6.5 Task Experience

- Login to DAMP with your account
- Navigating to the desired marketplace and Selection of game
- Payment method selected (paypal/ cryptocurrency)
- Enter received game key into desired platform (Steam, Origin, etc)

1.6.6 Frequency of Use

A user would use our application whenever they want to purchase a game. This could occur daily, weekly, monthly or yearly depending on the frequency that they purchase games.

1.6.7 Key Interface Design Requirements That the Profile Suggests

- Web interface to facilitate the browsing and purchase of games
- Space on the interface for advertisements
- Integration with cryptocurrency and electronic payment systems
- Store pages in the marketplace for developers to host their games

1.7 USE CASE DIAGRAM

Our project has three main parties involved with the system: users, developers, and administrators. Figure 1 outlines the use case for each of these parties. Users and Developers will share all of the same functions as users; the difference being that if a user chooses to upload their own game, that then classifies them as a developer. Admins will consist of our project team, and will be responsible for upkeep of the limited backend functions, such as webpage maintenance and managing and monitoring storage pools.

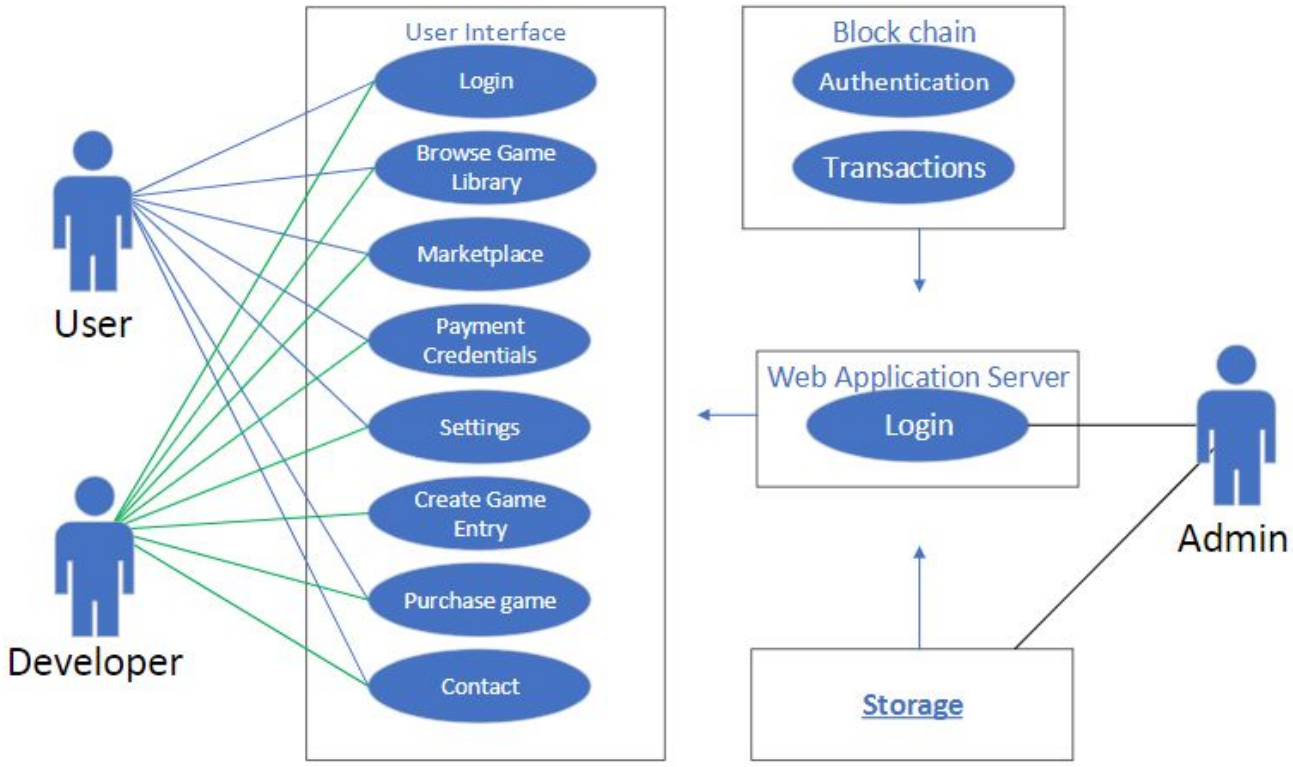

Figure 1: Use Case Diagram

2. PROJECT MANAGEMENT

2.1 Budget

Table 1: Project Budget presents the estimated cost for the first year of development for our project, DAMP. The cost is based off of 3 developers each working an average of 15 hours a week on the development, Gas cost used for Smart Contracts and deployment on the Ethereum Network. Smart Contract calls are required for each game sale being made with an estimation of around 5000 sales within the first year with the average gas cost being six cents each. Throughout the development, a test network was used which cost us nothing to work on and develop.

Table 1: Project Budget

ITEM	UNIT #	COST	TOTAL
Labor	1620 Hours	$20	$32,400
Smart Contract Deployment Gas Cost		$1	$1
Smart Contract Calls Gas Cost	5000	$.06	$300
TOTAL			$32,701

2.2 Objectives/Deliverables

Create digital asset management platform via blockchain capable of performing the operations of a modern digital distribution platform without pinning high platform operating costs on game developers and consumers. DAMP will imitate the marketplace principles available on major gaming distribution platforms alongside offering game download hosting.

Features:

- DAMP will offer a user friendly interface and cross-platform capabilities for both mobile and stationary devices.
- Developers will have the ability to create their own unique marketplace to house their games and allow for customers to buy from them directly
- Developers will have the option to host their own game storage or use our hosted storage
- All user identities will be transparent through DAMP
- Initial support for Ethereum currency with support for other major cryptocurrencies to follow long term
- Incentivizing users to host content by offering a percentage of all mined transactions to them

2.3 Project Schedule

Figure 2: Fall Project Schedule & Gantt Chart is our Fall project schedule with milestones listed.

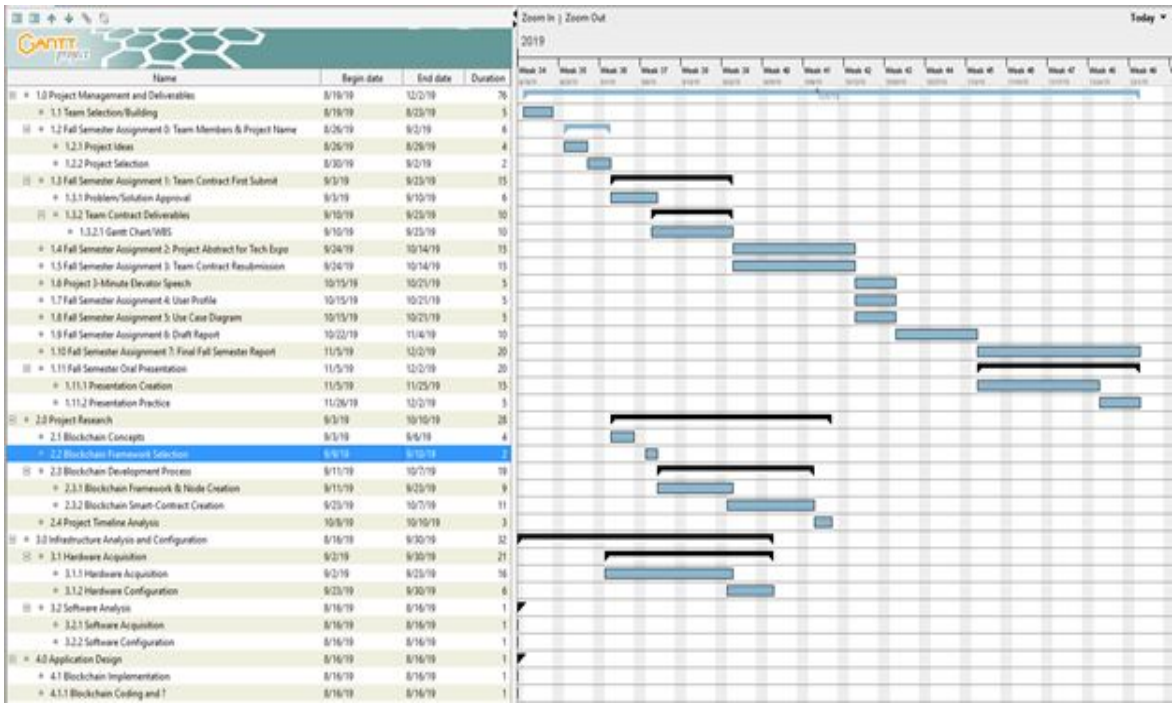

Figure 2. Fall Project Schedule Gantt Chart

Figure 3: Spring Project Schedule and Gantt Chart is our Spring project schedule with the major milestones listed for the entire year.

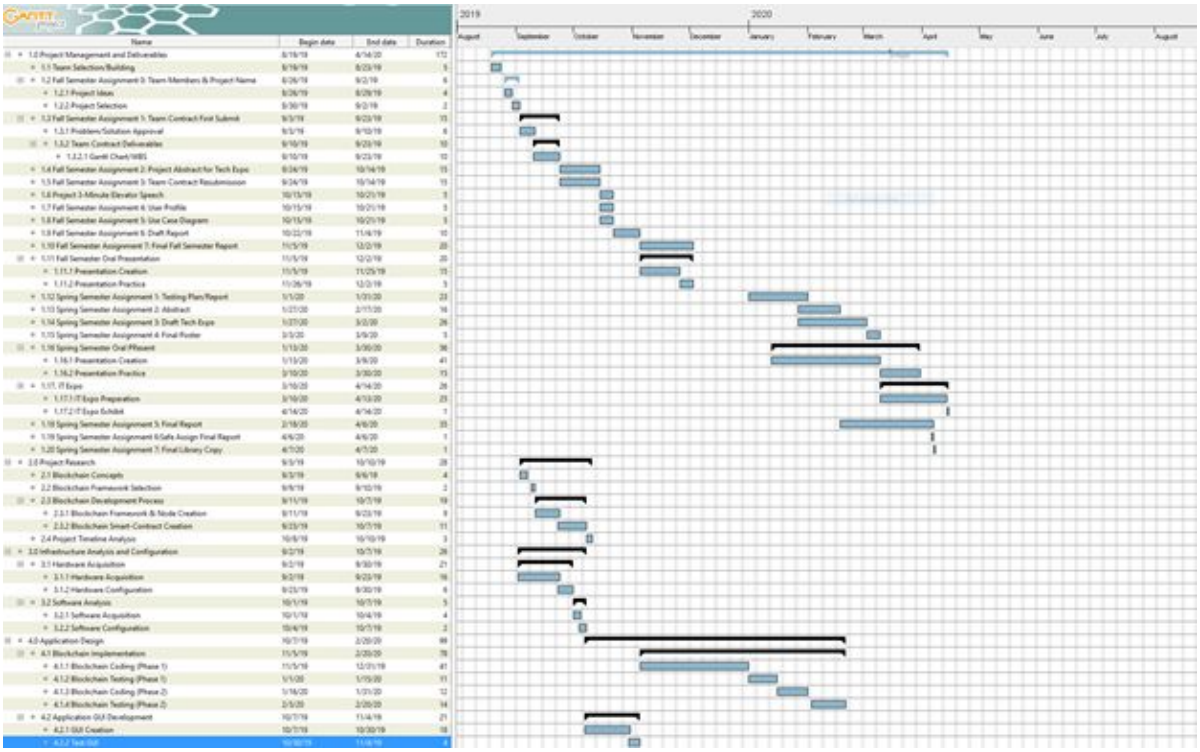

Figure 3. Spring Project Schedule Gantt Chart

3. Technical Elements

3.1 Network

DAMP will use the Ethereum blockchain to publish and process each transaction. Our network will therefore be node-based, meaning each computer on the network does part of the contract processing. We will also have a network of decentralized storage that will allow users and developers to store and download their games similar to a torrent. In addition, there will be a small, static storage pool that will be maintained by the administrators for backend purposes.

3.2 Application

Our web application will be written in Python using the Bootstrap library that will communicate with the Ethereum Blockchain. Our front end will communicate with the smart contracts on the Ethereum network. These contracts will record a token when a game is purchased which will grant the user access to their game. Users will be able to download the game from any source, the game will require the token purchased in order to work.

Figure 4 below, is the Login page where a user can sign in or register for an account that they will use to browse the marketplace and purchase games.

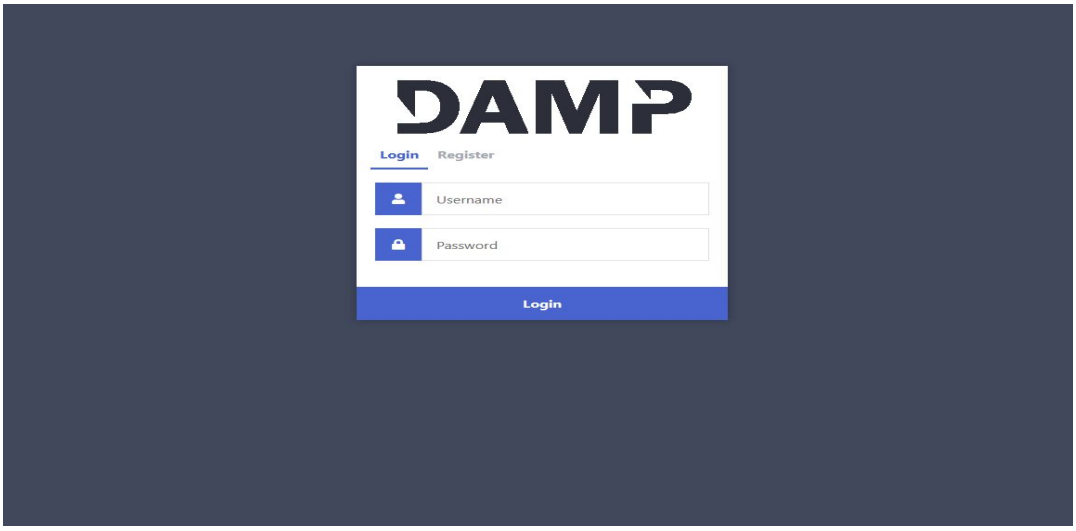

Figure 4. Website Login Page

Figure 5 below, a user can browse the Indie Developers marketplace where they are able to select a specific game to get more information about and have the option to purchase.

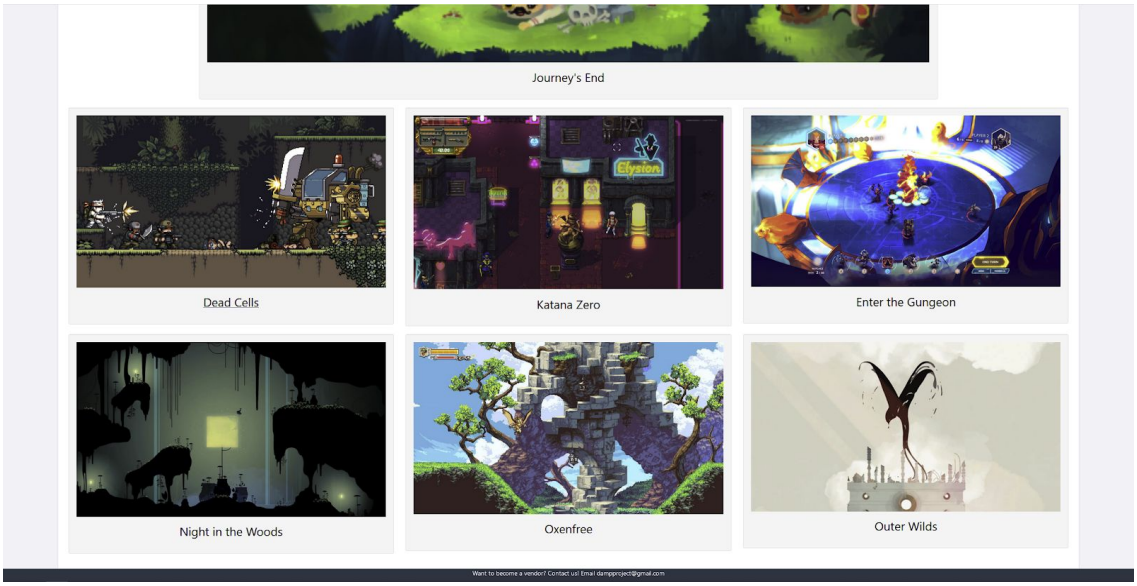

Figure 5. Game Marketplace

Figure 6 is the page of a specific game that a user would be able to navigate from the marketplace where they can look at a game's details and purchase with the currency of their choice.

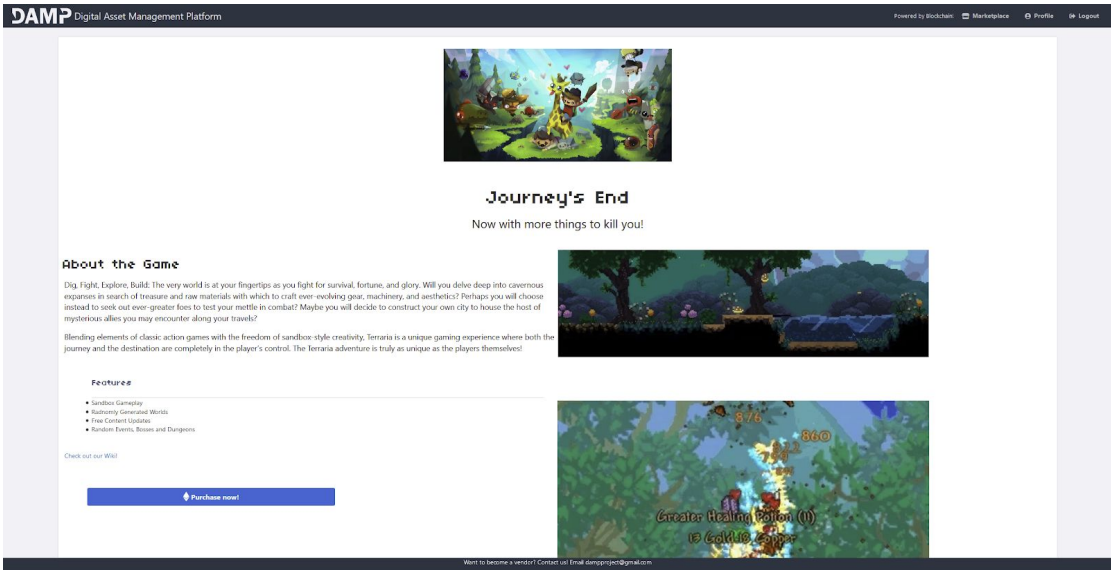

Figure 6. Game Homepage

3.3 Security

Blockchain at its core is inherently secure due to uses of consensus and immutability. Interactions with the blockchain are written and committed so that the whole community of users has visible access. This kind of visibility makes it very hard to alter the blockchain in an attempt to steal a transaction. So Blockchain is a safe technology that doesn't need much additional oversight for security.

Our storage possibility will sit behind the web page login and will only be accessible for download privileges for customers. Admins on the back end will handle the upload of games to storage directly from developers making storage roundabout secure. Our webpage is still a developing piece when it comes to user login and blockchain interactions. As such, we are not able to comment on how security will work with it yet. We first need to make some decisions concerning the use and storage of usernames/passwords and where these will come from.

3.4 Technical Diagram

Our solution consists of several key components that will allow our users to browse and purchase games from the marketplace. The web server will be the central component to this, acting as an intermediary between the users and the backend, such as storage and blockchain components. On the backend, we will be using the Ethereum Virtual Machine to process our smart contracts and will have a Peer to Peer storage network in place to deliver content to our users.

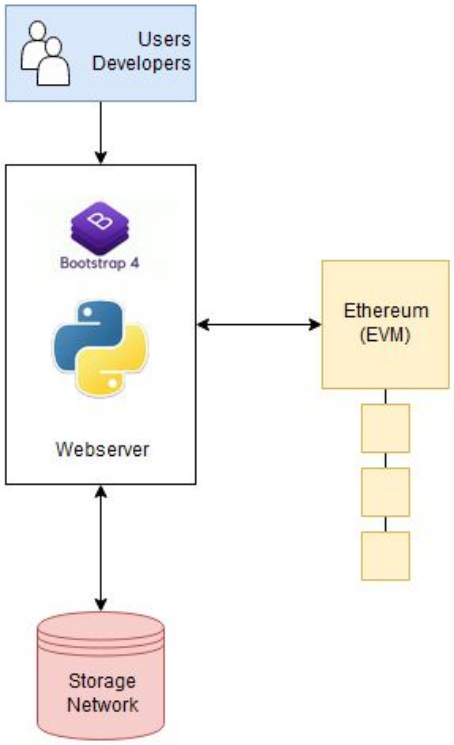

Figure 7: Technical Diagram

3.4.1 Test Environment

Our test environment is currently a private blockchain that our team is using to test smart contracts and transactions. These contracts are written in Solidity, the language on which Ethereum and all other smart contracts are built. Once they're compiled, they're executed on our private Ethereum blockchain, which is hosted with Ganache and interfaced with Truffle. Finally, Metamask allows us to create accounts that can monitor Ethereum wallets to verify transactions are initialized.

Figure 8: Test Environment

Below are **Figures 9 and 10** which are screenshots from our test environment. The left image shows what a freshly created private blockchain looks like, along with the test accounts used to complete the transactions. The screenshot on the right shows what the blockchain looks like with a basic transaction and a smart contract fulfilled.

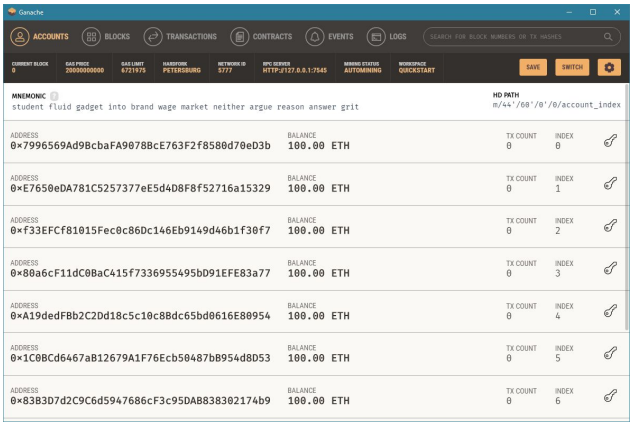

Figure 9. Ganache Accounts

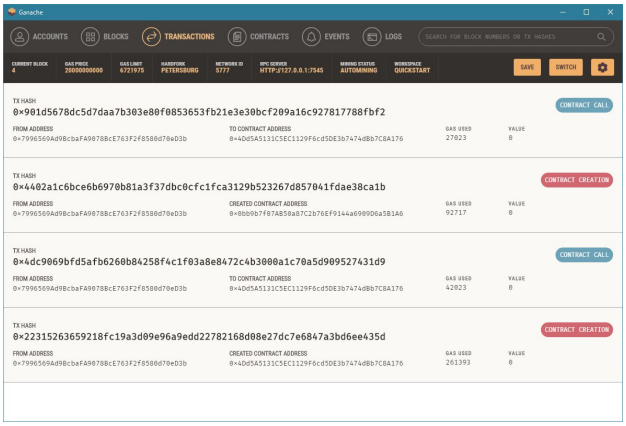

Figure 10. Ganache Transactions

Figure 11 below is a screenshot that shows a few examples of the code behind the transactions. The config file (left window) is written in Javascript and tells Truffle, the smart contract framework, to connect to the localhost on port 7545, which is where it finds Ganache. The right window shows the migration contract, which is the first contract written to the blockchain. Below in the terminal window, you can see the output from running the 'migrate' command, which initializes all of the contracts and, in our case, processes them immediately on our private blockchain for the current gas price.

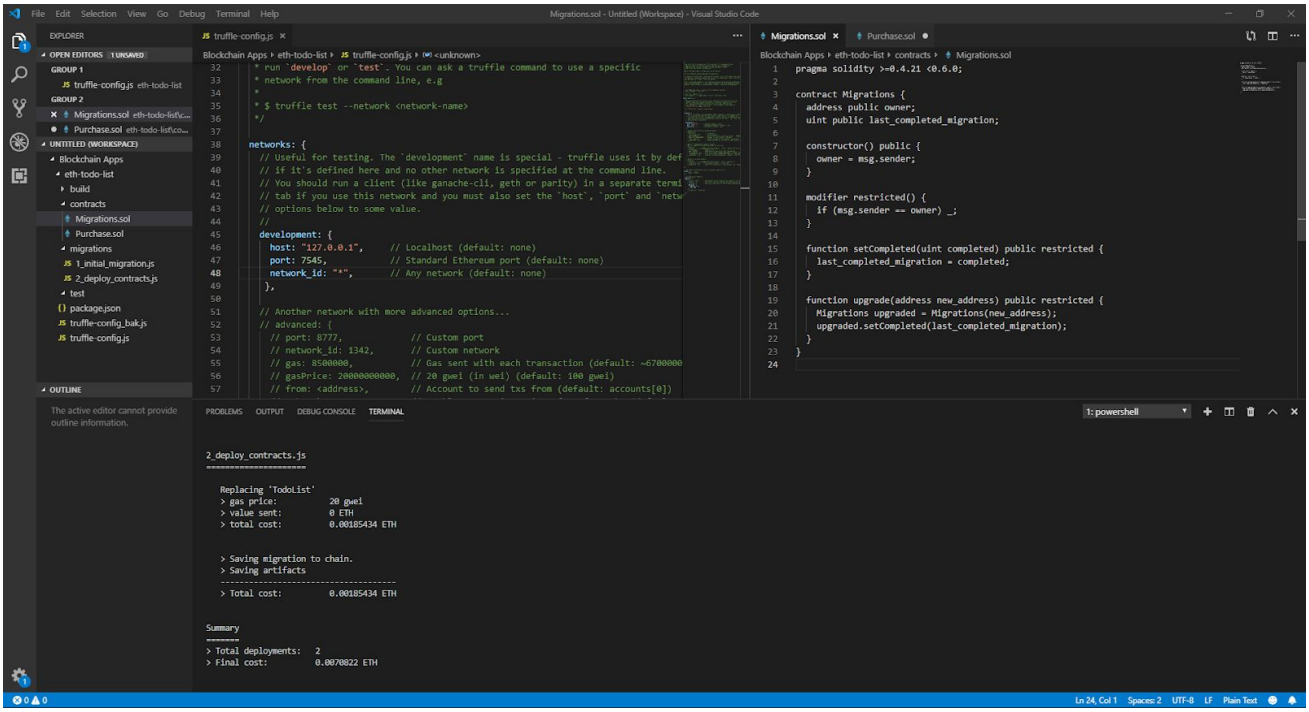

Figure 11. Truffle Application

4. TEST PLAN

4.1 Overview and Methodology

We will be utilizing two different testing approaches to audit our project.

- The first will be a step-by-step quality assurance test to verify the functionality given certain test cases.
- The second will be on the fly as the development process continues(extemporary). We will report any bugs/issues as we discover them.

Our project consists of two primary components, each of which will require an in depth level of testing. As such, we will be alternating testing based on who created the component. For example, Dan will test Austin's frontend web page interface, and Austin will test the backend functionality. This is done to ensure that someone besides the developer is also testing the code to ensure that nothing is overlooked due to being too close to the component in question.

Our team has compiled a protocol for testing that we will be using for our manual QA test cases:

1. Test cases will be created ad-hoc by both the front and backend developers
2. QA tests will be run by the alternate developer (Frontend Dev tests the backend, Backend Dev tests the frontend) and the Lead QA Engineer
3. Lead QA Engineer will document each test and record the outcome.
    a. Successful tests will be marked as complete
    b. Failed tests will be marked as a bug

4. The testing process will be repeated as necessary across all test cases until a satisfactory level of functionality is achieved

Our test cases will be designed to encompass each of the core functions that any consumer of our platform may perform. These actions compound many sub functions split between the front and back end, so a mixture of our two methods of testing will be required. Extemporary testing of smaller features as code is written will improve and catalyze our QA testing down the road.

4.2 Scope of Testing

The testing done for DAMP will include all intended features built within our application. This will include testing for both the End User Role and Developer selling games. In this application testing, the goal will be to run through a start to finish process from any possible users who would interact with the application.

Work Item Type	User Type	Title	State
Test Case	Customer/Game Developer	User Login	Ready
Test Case	Customer	Navigate to Game Library	Ready
Test Case	Customer	Navigate to Marketplace	Ready
Test Case	Customer	Purchase Game and Simulate Payment	Ready
Test Case	Customer	Open Settings	Ready
Test Case	Game Developer	Test Contact Us Link	Ready
Test Case	System Admin	Test Login to Web App Server/Ensure site properly hosted	Ready

Figure 12. List of all Test Cases

4.3 Logging Test and Procedures

As mentioned, we will have two testing methods being step-by-step and on the fly (extemporary). Any reported bug and issues will be documented within our “Bugs” spreadsheet, located within Google Docs that we regularly check. As an additional measure, the event will be communicated via Telegram, our group’s primary messaging mobile app.

Once a bug is identified, we will determine if it falls under the front-end web page interface or backend functionality. From there it will be claimed accordingly, investigated and documented. Each test case will be performed three times, once by each team member to ensure consistency and avoid any biassies.

Manual testing will be done on an individual basis per each team member’s own discretion. For reporting bugs, the same process as listed above will be followed. There is no timeline for this testing, as we feel confident to know how much will be needed for our website.

4.4 Test Results

Work Item Type	User Type	Title	Expected output	Result	Date Tested
Test Case	Customer/Game Developer	User Login	User should be able to log in to the website and create a new account if they do not have one.	Successful	2/16/20
Test Case	Customer	Navigate to Game Library	User should be able to view their profile, allowing them to view their game keys and account info.	Successful	2/16/20
Test Case	Customer	Navigate to Marketplace	User should be able to view and navigate the games marketplace, allowing them to access game pages for each game.	Successful	2/16/20
Test Case	Customer	Purchase Game and Simulate Payment	User should be able to access all game pages and click the purchase button, executing a payment on the blockchain.	Successful	3/12/20
Test Case	Game Developer	Test Contact Us Link	User should be able to click the link to contact the developers via email and phone.	Successful	2/17/20
Test Case	System Admin	Test Login to Web App Server/Ensure site properly hosted	Admin is able to log in to the web server and manage the site.	Successful	2/19/20

Figure 13. Test Results

5. CONCLUSION

5.1 Fall Semester 2019

Our group's main focus for the project for Fall semester has been teaching ourselves the basics of a Blockchain network and learning of all the different types of currencies available so that we could pick which would be best suited for our idea. We have also learned the beginnings of a coding language called Solidity which is used to develop a working blockchain on Ethereum. Thus far we have achieved the creation of a sample blockchain with operating smart contracts so that we may test the exchange process for our customer/developer interaction. Going forward our group aims to develop a web page to interface with the blockchain for users to log into. We also look to find ways to minimize the initial cost of the ethereum network for smooth and cheaper deployment in the future.

5.2 Spring Semester 2020

For the spring semester, our group focused primarily on the web page development and interaction sequence with our private blockchain. We began with our wireframe that we finalized the end of Fall semester, and developed it using a Bootstrap/Python language. We focused on simple user interaction that allowed quick browsing and easy purchase options. Our web page interacts directly with the blockchain when purchases occur, triggering a smart contract to run the appropriate terms and create the transaction into the blockchain's ledger. We initially struggled with achieving this interaction but managed to resolve it on our private blockchain. We solidified our Blockchain development skills and increased our understanding of Bootstrap/Python web page development. Our group has come to the realization that we would need significant adjustments for the Blockchain interaction process with our web page, but we could certainly achieve the desired results given more time and motivation.

5.3 Problems Encountered and Future Recommendations

Throughout the course of our project, we encountered a few large issues that caused us to rethink our project. First and foremost, we ran into immediate issues when, after deciding on our project idea, we had no clue which blockchain would serve our use case best. Setting out to make a transaction ledger, it seemed that we had several platforms to choose from. Originally, we considered using Hyperledger, an IBM project that was rigid but powerful, and seemed to be a good option for developing our project. However, after more research, we switched to using Ethereum, the open source blockchain platform used by thousands of small projects with a very large support base. The benefits of having a more widely

supported platform made developing our project much easier, and gave us more flexibility to develop our asset management platform in a way that worked best for us.

Another issue we came across was the list of features that we wanted to realistically achieve within the course of these two semesters. Knowing that we were going to compete with larger gaming platforms such as Steam, Epic, and EA, this required us to provide a specific list of features regardless. This forced us to narrow down the features to those that most other platforms provide such as buying a game and being able to reference past purchases on top of having a worked game marketplace.

Going forward we would want to tackle a more definitive means of verify keys of games after transactions. Key verification was an oversight on our end when we first conceived the project. It was only till we were nearly done that the concept was brought to our attention and by then we had to accept that it was going to be a missing component. Aside from key verification, the future of this project would've been analyzing the changes we would have to make to bring our private blockchain into a public blockchain. Here we would have to tighten down on security on our end and could begin to enact the Peer to Peer storage network we had mentioned in our project's potential benefits.

APPENDIX A. ADDITIONAL INFORMATION

The team contacted the following professionals for clarification and additional insight.

1. Bo Vykhovanyuk, Advisor
2. Rashmi Kawasaki (84.51°), Coworker and advisor

APPENDIX B. REFERENCES

Fenlon, Wes. “Steam Now Has 90 Million Monthly Users.” *Pcgamer*, PC Gamer, 14 Jan. 2019, www.pcgamer.com/steam-now-has-90-million-monthly-users/

Sinclair, Brendan. “Just 6% of Devs Say Steam Earns Its 30% Cut - Survey.” GamesIndustry.biz, January 24, 2019. https://www.gamesindustry.biz/articles/2019-01-24-just-6-percent-of-devs-say-steam-earns-its-30-percent-cut-survey. (Sinclair 2019)

Vincent, Brittany. “Valve Introduces New Revenue Split Changes For Steam Sales.” Variety, December 3, 2018. https://variety.com/2018/gaming/news/valve-revenue-split-changes-1203078700/. (Vincent 2018)