Final Project Report

Diego Cifuentes
University of Oklahoma
CS 5833: Blockchains & Cryptocurrencies - Spring 2024
May 5^{th}, 2024

https://github.com/diego2201/MarketPlace
https://diego2201.github.io/MarketPlace/

Objective:

The purpose of this project is to develop a web based decentralized Ethereum marketplace interface, which enables the users to interact with a smart contract to list and purchase items securely. Leveraging the Ethereum network, the DApp allows the users to conduct transactions directly from their browsers (Google Chrome) using MetaMask for their wallet management and approval. The interface provides functionalities to list new items, display all items on the marketplace, and purchase items. Each transaction, whether listing or buying, is recorded on the blockchain, ensuring transparency and security. The purpose of the project is to simplify the process of buying and selling items on the blockchain, making it accessible and manageable through a user-friendly web interface.

Important Considerations:

Due to a few of the decisions taken, which will be explained more in depth in the project design section, there are a few things that need to be taken into account when using/testing the website.

The main consideration here is, the fact that you need to have MetaMask installed in order to use the website. If MetaMask is not installed, you will not see any listings and not be able to list or purchase item. The work around here was to at least use Infura to be able to show the listings without the user needing to have MetaMask installed, but due to a few issues that arose when using this, I deemed that it was not worth it and provided a mediocre service to the website users.

Another consideration here is the speed of the writing and reading of the blockchain. I have implemented a function that is called about every 8-10 seconds to check if there is any changes in the listings (the boolean for if the item is sold has changed from false to true) or any new items listed. If a change/new listing has been detected, then the page will refresh. I find that 8-10 seconds is the best time spand, slow enough allowing the program to not slow down with rapid checks but also fast enough to detect a change and display it to the user in a timely manner. However, there may be some hiccups, because of this it may just be easier to completely refresh the webpage to display any changes!

Another quirk of the website is that when you are listings something it asks for the list price in Wei and then it displays it in Eth once it has been pushed to the contract.

The contract itself also has a quirk, where it begins the listings at 0 and then counts up instead of starting from 1.

The last main consideration here is that, if you have more than one MetaMask account that you wish to use, you will have to manually change your account through the browser extension. The website will recognize this, but may not update the variables that are displayed to use (ie. Account address and balance). To fix this all you need to do is click on the connect button again and these values should refresh.

Project Design:

When getting started with this project, we were given a rough outline of how the website should work and what tools could be useful. However, as time progressed, as I read more into the given tools, and with troubleshooting the website, a few decisions were made to the project design in order to make it a seamless and reliable experience.

One of the biggest project designs taken here was the removal of Inufra for the Web3 functionalities in favor of relying completely on MetaMask. This comes down to a few reasons. One big reason here is that Infura only allowed for reading from the blockchain and writing, unless using a hardcoded private key (not very secure), could not perform any writing and/or signing of any data. Despite all of this, Infura did offer one benefit over using MetaMask. This was the fact the website would be able to read from the blockchain (showing all of the items listings). However, once there were about 5 or more listings, Infura would throw an error. This error ended up being because the website was sending too many requests in a short time period to the Infura server. To fix this, I implemented a wait function, which would wait about 2 seconds between calls in order to try and help slow down the requests. While this did work, it would appear to someone in a rush that the website was not working. It would also make the auto refreshing of the page laggy and unprofessional, especially in the case where multiple transactions occurred at once.

The first thing I did for this project was design the Smart Contract and test it using the Remix IDE. There I ran through a few iterations, believing that there was an error as I would not see any changes in either the seller or buyers balance. Despite the IDE having a tab to change from Wei, to Eth, and so forth, I found that it did not work reliably. Because of this I had to keep entering my value in wei. So if I wanted to list something as 1.2, I had to enter 1200000000000000000 into the field. This quirk then carried over to the front end as well.

At the start of this project, I wanted to implement a sort of drop down menu where if the user had multiple MetaMask accounts linked they could select from those. However, I was not quite able to implement that, so I just let it be that the user can manually switch accounts.

For this project, I decided to use basic JavaScript, HTML, and CSS for the website. This is for a few reasons. The first is that, I am unfamiliar with any frameworks and/or how to use them. Due to time constraints I believed it would be faster to just uses basic JS, HTML, and CSS instead of delving into a framework and understanding how the interactions work. Another reason is that using these for Github pages would also be efficient. That is because Github pages is a free platform used to host any static websites. Using a framework would push me to use Google Cloud Platform to host the website, however, also due to lack of experience and the time constraints I believe that Github pages would be a good platform to use to host this website.

Testing:

I tested this project in various ways. I tried to break it down and testing each section individually first, and then testing them once they all came together.

For the Solidity contract, I used the debugger tool in the Remix IDE. This is because at the beginning the smart contract I wrote had a bunch of errors. The learning curve was steep, but I eventually found tutorials and forums online that helped me understand and work my way around the debugger.

I also set up unit testing on the Remix IDE to test the various functionalities of the smart contract.

Challenges Faced:

The main challenge that I had at the beginning was just knowing where to start. At the beginning I felt like it was so open ended, that I began to over complicate things and focus on aspects of the project that could be simplified or just weren't worth he time. For example, I wanted to over design the webpage and make it look and function similarity to other e-commerce websites like Amazon. I also wanted to add a theme to the websites, like focusing on the first 151 Pokemon and being able to list and buy their cards along with their respective images. However, again, due to time constraints and complexities I eventually decided against this.

One challenge faced during this project was getting everything actually working together. Created the smart contract was a fairly straightforward task, especially with all of the support and tutorial online of other creators making similar marketplaces or bidding sites. Making a simple website based off of HTML and CSS was also not too bad, even considering that I do not have much experience in this topic. However, once the website and the smart contract were created, I now needed to get them working together, ie. allowing the user to use the website to interact with the smart contract. I did find that the main issue, was actually figuring out the commands that needed the user to sign off on. Things such as listing a new item or purchasing something, however, reading, such as listing all of the items currently on the smart contract, was fairly easy and straight forward.

Another big challenge was figuring out which resource was really worth the time and implementation into the project. The biggest suspect here is Infura. While, it was recommended to use, after a few days of tinkering with the project and figuring out various other functions and issues, I find that using Infura was more time than it was worth, and even made the code run slower than it should. This caused it to seem that the website was not truly working and/or offered a bad user experience due to the wait. After some consideration, I took the decision to move away from this and solely rely on MetaMask for both read/write operations to the Blockchain.

Finding reputable and reliable sources to help build this project was also another challenge. While there were a fair share of example projects, similar to this Marketplace, I found that they were out of date. I tried to follow along, but certain tools they used were deprecated or outright unavailable, smart contracts were written in an older version of the Solidity language, certain command line commands no longer worked, and so forth. Regardless, these still helped to lay the foundation for the logic that I would use for the project.

Demo:

Here is how the website looks and a link to a Demo:

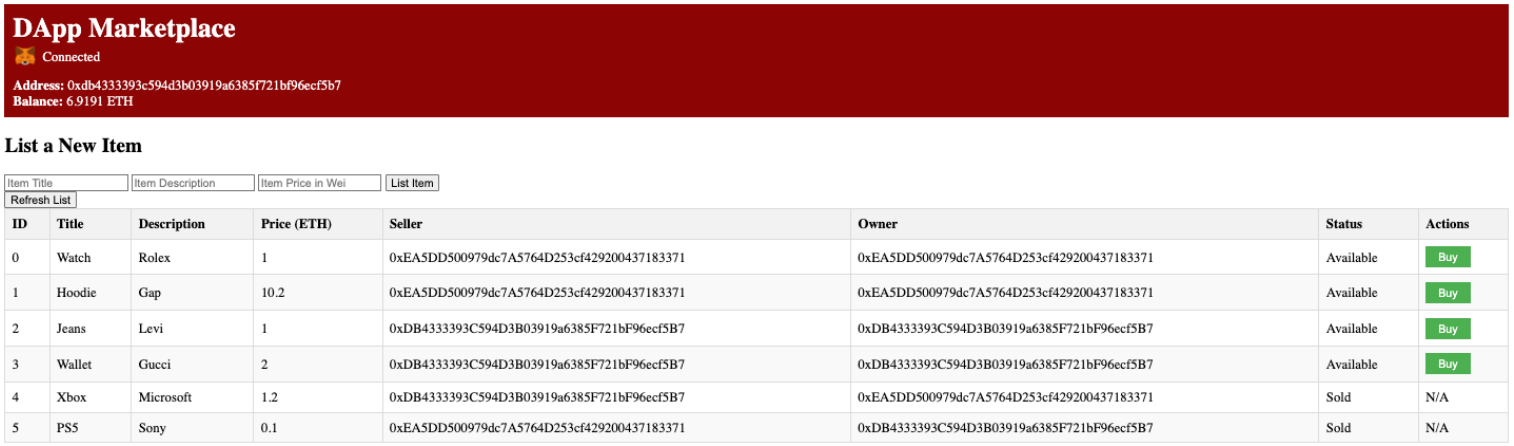

FIGURE 1. https://mymedia.ou.edu/media/t/1_mhivmp72

I also provided a demo to show that the transaction hashes can be verified on EtherScan:
https://mymedia.ou.edu/media/t/1_niu2vz7i

Here are the important links as well:
https://github.com/diego2201/MarketPlace
https://diego2201.github.io/MarketPlace/