AI Cannabis using Blockchain for transport analysis in the Metaverse

Aishwarya Sharma, Anviksha Sharma & Vincent Davidson*

Abstract. With the continuously evolving digital technology, the demands of the shipping industry have transformed as well. In recent years, blockchain has garnered a lot of attention in the industry for its wide range of applications in transportation, supply chain, and shipping. In this project, we aim to enable the use of blockchain with Artificial Intelligence (AI) to implement a technology for users to make purchases in the Metaverse and further use Machine Learning (ML) to predict the delay in the delivery of the shipment.

Keywords. Artificial Intelligence (AI); Blockchain; Metaverse; prediction; Supply-chain; shipping

1. Introduction

With the introduction of several new trade paradigms, the increasing demand for new propositions in production and distribution due to globalization will continue to change the operational standards of shipping.[10] Our project aims to use blockchain with AI for making delivery time related predictions involving delays, hence providing capabilities to individual cannabis vendors to boost sales and revenue.

There are already companies using AI and machine learning technology to optimize the cannabis market growth from seed to sale/delivery. We plan to evaluate AR technology for the user to make purchases within the Metaverse. The delivery services can be evaluated by:

1. Transport analysis: Evaluation of real time efficient routes for drivers and suppliers. Being able to forecast arrivals and deliveries (using AI) from within the network. This allows for a higher performing application output and will assist in streamlining the real-time operations that this adds to the design.

2. Blockchain Component:

   (a) It comprises of using timestamps of deliveries and verifying destinations.

   (b) It will also be used for the transfer of money from user to supplier. A successful transaction will only take place after the product is successfully delivered to the user.

   (c) It helps make predictions regarding delays in shipment delivery.

2. Background and Related Work

As intelligent transportation systems have started emerging, they are finding applications in a lot of areas namely internet of things, self driving vehicles, data analysis etc , while this is a great technological advancement, it is making them extremely vulnerable to security issues.

The paper "Towards Blockchain-based Intelligent Transportation Systems" by Yong Yuan et al states a novel approach based on the decentralized architecture and distributed computing

* Authors: Aishwarya Sharma, University of Florida (aishwarya.sharma@ufl.edu); Anviksha Sharma, University of Florida (sharma.anviksha@ufl.edu) & Vincent Davidson, University of Florida (vincent.davidson@ufl.edu).

paradigm using blockchain to handle the two main security issues faced by intelligent transportation (ITS) systems which are 1. The need to secure centralized platforms or cloud services used to store and process data that is collected for modern day applications like self driven vehicles, data analysis etc and 2. Lack of necessary mutual trust among ITS entities which results in , money and assets not being able to "flow" from one entity to another smoothly and directly without trusted intermediaries, resulting in hierarchical structures, diversified mechanisms and increased social complexity of ITSs. The paper also mentions a case study of one of the most successful application scenarios of Blockchain-based Intelligent Transportation (B2ITS) in La'zooz which is publicised as "blockchained version of Uber".

Apart from this, the paper "AI-Powered Blockchain - A Decentralized Secure Multiparty Computation Protocol for IoV" by Gunasekaran Raja et al, introduces the concept of intelligent contracts as opposed to smart contracts to tackle the current problem with the blockchain technology which is that it can never create mass adoption because of its complexity. They suggest AI storing highly sensitive and large amount of data collected by ITS in blockchain which is then provided to the AI engine to make statistical and analytical decisions. The approach proposed by the paper for their the decentralized ethereum application also uses an autocoded intelligent contract which tremendously improves its efficiency. They use NLP to autocode the smart contract and Naive Bayes classifier to predict the tag for the each user request.

There is also enough research on the evolution of the retail space from traditional to electronic to metaverse in which they highlight the key challenges and opportunities faced by traditional retailers, e-retailers and metaverse retailers in the paper "Retail spatial evolution: paving the way from traditional to metaverse retailing" by Michael Bourlakis et al.

Though there is enough research on how to secure transportation data already collected using blockchain and upcoming AR retailing platforms. There isn't much research on how to apply B2ITS to retail in an AR environment like metaverse. Our proposal is novel because we are trying to fill the gap between existing research in blockchain based Intelligent transportation system and retail in AR environments by trying to find ways secure Cannabis sale and transport in metaverse by applying blockchain and AI to it.

3. Metaverse: Design & Working

Metaverse is defined as a network of 3D virtual worlds with a focus on social connection.[13][14][15] It has frequently been described as a theoretical iteration of the Internet into one universal virtual world that is aided by the usage of virtual (VR) and augmented reality(AR) headsets.[13][16] The term originated in 1992 in a science fiction novel called Snow Crash. The definition of "metaverse" has evolved and extended over time. Various types of metaverses have been developed for use as a platform for virtual worlds such as Second Life.[17] The creator Philip Rosedale described Metaverse as a three-dimensional internet inhabited by living people. Some metaverse variants virtual and physical space and virtual economic integration.[13] The development of metaverse has been linked to the rapidly advancing virtual reality technology.[18][19][20] Increase in work productivity, e-commerce, real-estates, interactive work environments are some of the projected applications of the metaverse.

[1]

[1] GitHub link to the code: https://github.com/aishwaryasharmaccoew/TrackAnalize

3.1. Technology

3.1.1. Hardware

Metaverse access points include general-purpose computers and smartphones, as well as augmented reality (AR), mixed reality, virtual reality (VR), and virtual world technologies. [19] Reliance on VR technology limited the development of the Metaverse and thus has not been widely adopted. [19] High quality graphics and portability are lacking due to the limitations of portable hardware and the need to balance cost and design. Lightweight wireless headsets struggle to achieve the pixel density of the Retina display required for visual immersion [21], but the more powerful models are wired and often bulky. Current hardware development focuses on overcoming the limitations of VR headsets and sensors and increasing the immersiveness of tactile technology. [22]

3.1.2. Software

The standardized technical specifications for Metaverse implementations have not been widely adopted, and existing implementations rely primarily on proprietary techniques. Interoperability is an important concern in Metaverse development, driven by transparency and privacy concerns. [23] There were several standardization projects for virtual environments.

4. Issues to be addressed

The issue with blockchain-based persistence is that the chain could get far too big to upkeep and store all the data feasibly. The blockchain must also have some type of incentive structure. For blockchain-based persistence, there is a payment made to the miner and when the data is added to the chain, nodes are then paid to add the data on. Our work is both a research project and an engineering work. The research challenges and limitations have been discussed in depth in Section 9.

5. Proposed Approach & Plan

The problem of trust in retail on any virtual reality platform is huge, specially when it comes to retail. So by our solution, we planned to establish this trust by implementing the whole transaction on ethereum. Where the payments are made using smart contacts and the transaction history is stored on the blockchain for these two entities. For each transaction a new block gets added, so we are also using ethereum as our distributed data store. Once we have our tracking and transaction IDs for one shipment on blockchain we start to track the shipment using our python program.

For the case of this project planned to use USPS as our shipment service. After a fixed set of days which we call as our buffer period, if the receiver does not receive the product, our smart contract refunds the money back to the receiver. Else the sender gets the money. We define the buffer period as below.

$Bp(x) = E(x) + P(x)$

here, x is the current transaction in question. E(x) is a function that returns expected delivery time (In our case we will get it from USPS API {Refer to Appendix}). P(x) give the predicted delay for this transaction which we planned to get by fitting a prediction model on the data present on the ethereum chain for this transaction, which has start and end times of all previous transactions. Predicting P(x) is where we are trying to apply Machine learning on blockchain.

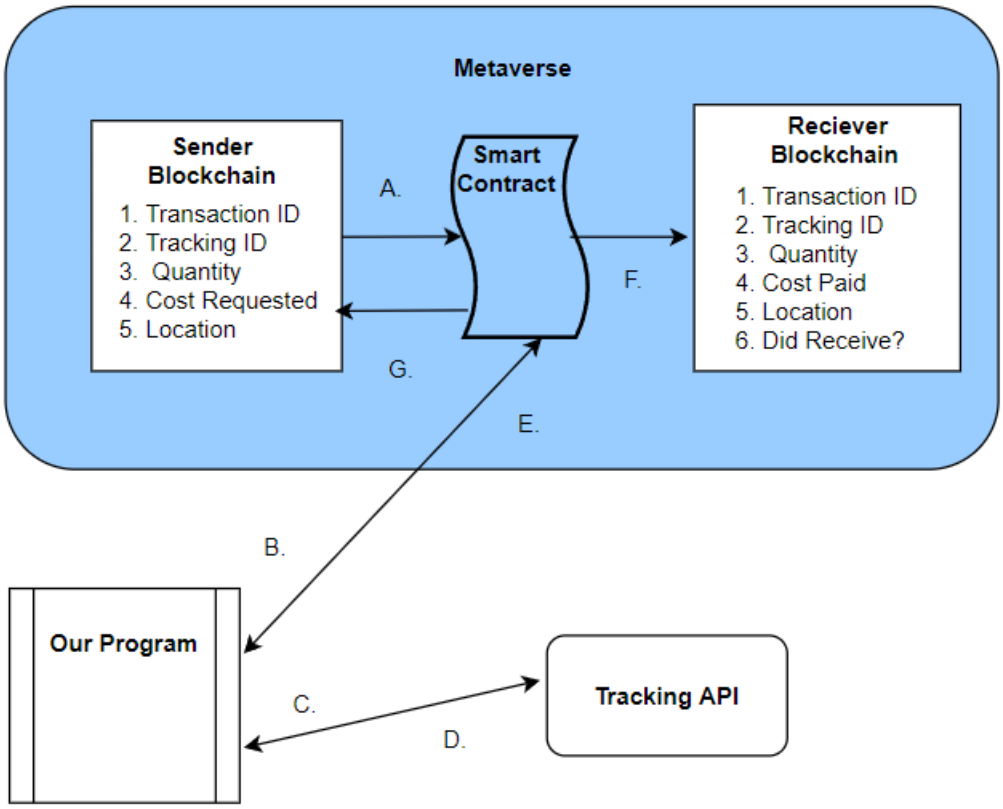

Figure 1. Architecture diagram

5.1. System Architecture

Fig 1. describes the working of our solution in the below mentioned series of steps:

a. The sender auctions cannabis in metaverse (AI aspect of our project).The highest bidder becomes the receiver and a transaction is initiated, which stores the transaction id, tracking id of the shipment, Quantity, Requested cost and delivery location on the blockchain.

b. Our smart contract deducts and stores the cost from the receivers wallet depending on the current highest bid.

c. Our python program then calls the tracking API of the delivery service using tracking ID entered by the sender.

d. The tracking API returns the information of the package as the response that includes the distance of the delivery route taken(in miles) and the time it took to deliver(in days).

e. Once the delivery response is received within the buffer time, our program calls the allow() or revert() functions of the smart contract.

f. If revert() in the smart contract is called due to additional delay or non confirmation of receipt, some portion of receivers money is refunded back to them

g. Else if the transaction goes through smoothly, the sender gets back the money.

6. Experimental Setup and methodology

To emulate our proposed system, we created a solidity project which had 3 smart contracts,Migrations, Auctions and UserCrud. Our contact was deployed using truffle and we used Ganache as the UI.

We also created a python program which is the driver code for our business logic that is off the blockchain. We also used python modules of web3 to be able to make rpc calls from our python program to our migrated contracts to read events.

Figure 2. Contracts

As this is a very complex problem statement, we had to simplify or methodology to be able to complete a substantial part in the course duration, which is as follows:

- The Auction contract, called first takes care of the bidding,refunding and holding money. Our UserCrud contract stores transaction information for each user in a self defined struct called UserStruct that has below variables.

    1. string userEmail; Stores the email Id of the sender for identification purposes
    2. string ruserEmail; Stores the email Id of the reciever for identification purposes
    3. uint price; Stores the final price of the cannabis sold.
    4. uint qtty; Stores the final quantity of the cannabis sold.
    5. string fromAddr; Stores geographical address of the sender, would help in prediction.
    6. string toAddr; Stores geographical address of the reciever, would help in prediction.
    7. uint time; Stores the total delay time it took to complete the deliveray
    8. uint index; Stores the tracking ID

```
PS C:\Users\Owner\Desktop\Courses\Blockchain\Projects\FinalProject> truffle compile

Compiling your contracts...
===========================
> Compiling .\contracts\Auction.sol
> Compiling .\contracts\Migrations.sol
> Compiling .\contracts\UserCrud.sol
> Compiling .\contracts\UserCrud.sol
> Artifacts written to C:\Users\Owner\Desktop\Courses\Blockchain\Projects\FinalProject\build\contracts
> Compiled successfully using:
   - solc: 0.5.0+commit.1d4f565a.Emscripten.clang
```

Figure 3. Successful contract compilation

- Due to time constraint as we were not able to integrate our usps api calls, we stored dummy data for all the above values, by calling the insertUser() function using truffle console.
- After this we installed and web3 for python and connected to our ganache server.

```
C:\Windows\system32>pip3 install web3
WARNING: Ignoring invalid distribution -ip (c:\python310\lib\site-packages)
WARNING: Ignoring invalid distribution -ip (c:\python310\lib\site-packages)
Collecting web3
  Using cached web3-5.29.0-py3-none-any.whl (500 kB)
Collecting eth-account<0.6.0,>=0.5.7
  Using cached eth_account-0.5.7-py3-none-any.whl (101 kB)
Collecting eth-hash[pycryptodome]<1.0.0,>=0.2.0
  Using cached eth_hash-0.3.2-py3-none-any.whl (8.8 kB)
Requirement already satisfied: requests<3.0.0,>=2.16.0 in c:\python310\lib\site-packages (from web3) (2.27.1)
Collecting websockets<10,>=9.1
  Using cached websockets-9.1.tar.gz (76 kB)
  Preparing metadata (setup.py) ... done
Requirement already satisfied: pywin32>=223 in c:\python310\lib\site-packages (from web3) (303)
Collecting jsonschema<5,>=3.2.0
  Using cached jsonschema-4.4.0-py3-none-any.whl (72 kB)
Collecting lru-dict<2.0.0,>=1.1.6
  Using cached lru-dict-1.1.7.tar.gz (10 kB)
  Preparing metadata (setup.py) ... done
Collecting protobuf<4,>=3.10.0
  Using cached protobuf-3.20.1-cp310-cp310-win_amd64.whl (903 kB)
Collecting hexbytes<1.0.0,>=0.1.0
  Using cached hexbytes-0.2.2-py3-none-any.whl (6.1 kB)
Collecting ipfshttpclient==0.8.0a2
  Using cached ipfshttpclient-0.8.0a2-py3-none-any.whl (82 kB)
Collecting eth-abi<3.0.0,>=2.0.0b6
  Using cached eth_abi-2.1.1-py3-none-any.whl (27 kB)
```

Figure 4. Web3 installation

- As soon as insertUser() is called, an event called LogNewUser is emitted.

```
truffle(ganache)> i.insertUser("pqr","pqr",1,1,"abc","pqr",50)
{
  tx: '0xcd9e7fb3de0cbb69d91cc8d5cfc08f39d815b6424e33717784c20592e569d8ca',
  receipt: {
    transactionHash: '0xcd9e7fb3de0cbb69d91cc8d5cfc08f39d815b6424e33717784c20592e569d8ca',
    transactionIndex: 0,
    blockHash: '0x1a5ef6677c8533e1fcea5880a19d9e5197e0d47dbcce5aa7cc181c119aaff682',
    blockNumber: 3,
    from: '0xe7de83429ca8417b846fad45ced79e56c59f8192',
    to: '0xaba9c71c4a6f113ccde0276c8121d132bd38339a',
    gasUsed: 23296,
    cumulativeGasUsed: 23296,
    contractAddress: null,
    logs: [],
    status: true,
    logsBloom: '0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000',
    rawLogs: []
  },
  logs: []
}
```

Figure 5. Call to insertUser

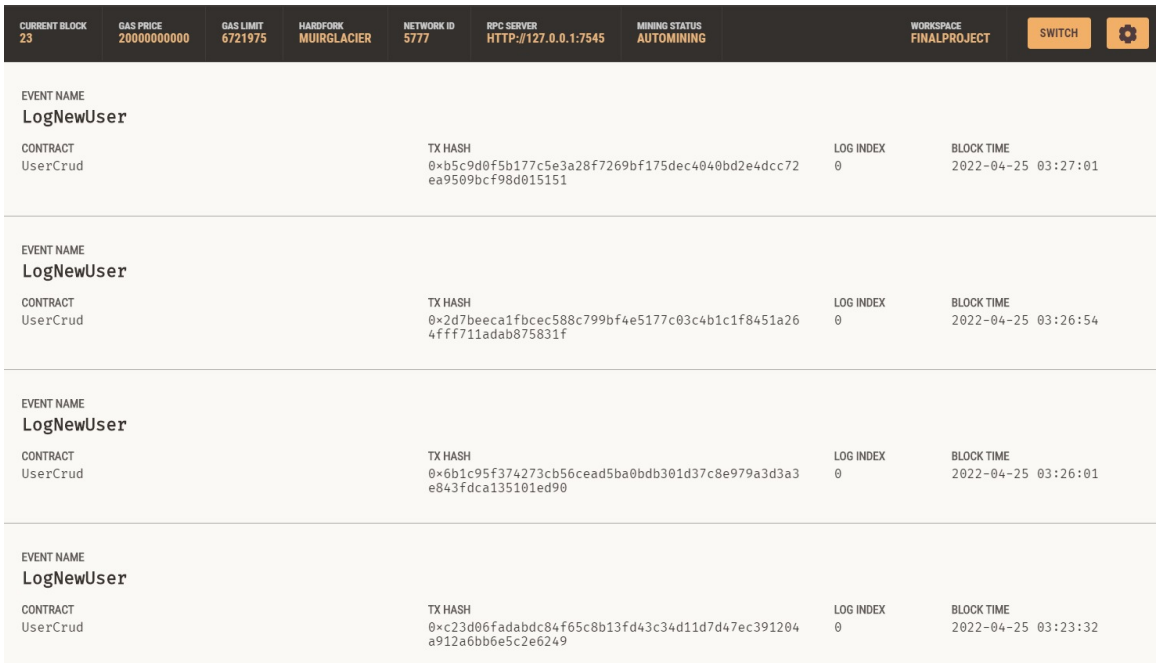

Figure 6. Emitted Event

- LogNewUser logs all values of UserStruct populated for that particular transaction.

Figure 7. Event as part of the function

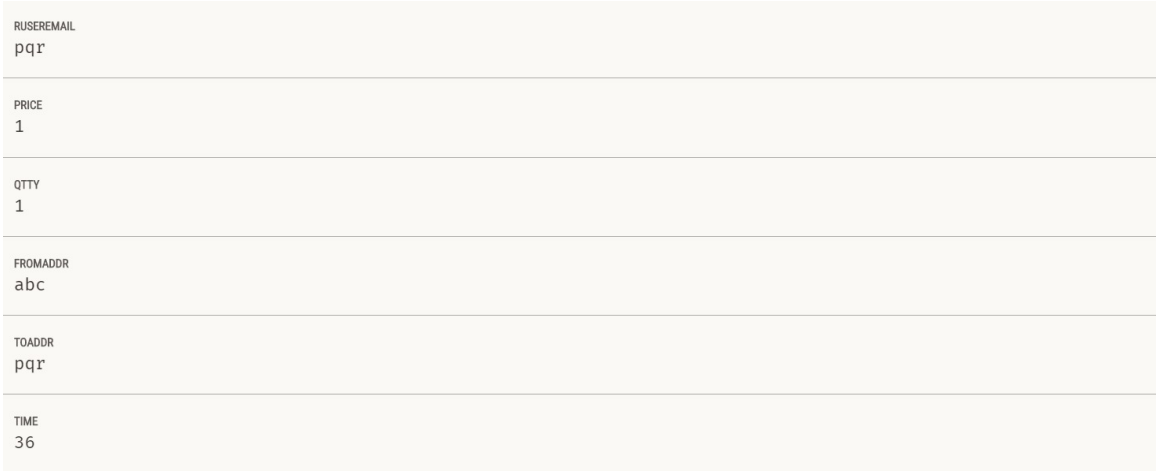

Figure 8. Event values

- Our python program called rpc-eth.py, has a contract variable initialized using the abi for UserCrud.json that we got after compiling our smart contract in solidity userCrud.sol.
- Using this contract variable, rpc-eth.py is able to filter all past events for a user by below command
  myfilter = contract.events.LogNewUser.createFilter(fromBlock= 0,toBlock= 'latest')

Figure 9. Gathering events in python

- These events are now stored in a csv file called vlaues.csv.

index	userEmail	ruserEmail	price	qtty	fromAddr	toAddr	time
0	'pqr'	'pqr'	50	3	'abc'	'pqr'	60
1	'pqr'	'pqr'	15	2	'abc'	'pqr'	20
2	'pqr'	'pqr'	20	1	'abc'	'pqr'	10
3	'pqr'	'pqr'	22	2	'abc'	'pqr'	20
4	'pqr'	'pqr'	10	1	'abc'	'pqr'	10
5	'pqr'	'pqr'	10	1	'abc'	'pqr'	10
6	'pqr'	'pqr'	18	1	'abc'	'pqr'	15
7	'pqr'	'pqr'	19	2	'abc'	'pqr'	20
8	'pqr'	'pqr'	24	2	'abc'	'pqr'	30
9	'pqr'	'pqr'	21	2	'abc'	'pqr'	20
10	'pqr'	'pqr'	25	3	'abc'	'pqr'	55

Figure 10. Plot of stored values before prediction

- We cleanse the stored data and retrieve the delay time value, for the scope of this experiment, we have hardcoded the distance in miles for each route in the variable uspsDistance.
- Next we wrote code to perform linear regression on delay time with respect to route distance to predict delay for a route.

7. Analysis

- From our analysis of the problem statement, we knew that the delays would be correlated to the route distances and so we decided to create a linear regression model that predicts delay as a function of accumulated route distances.

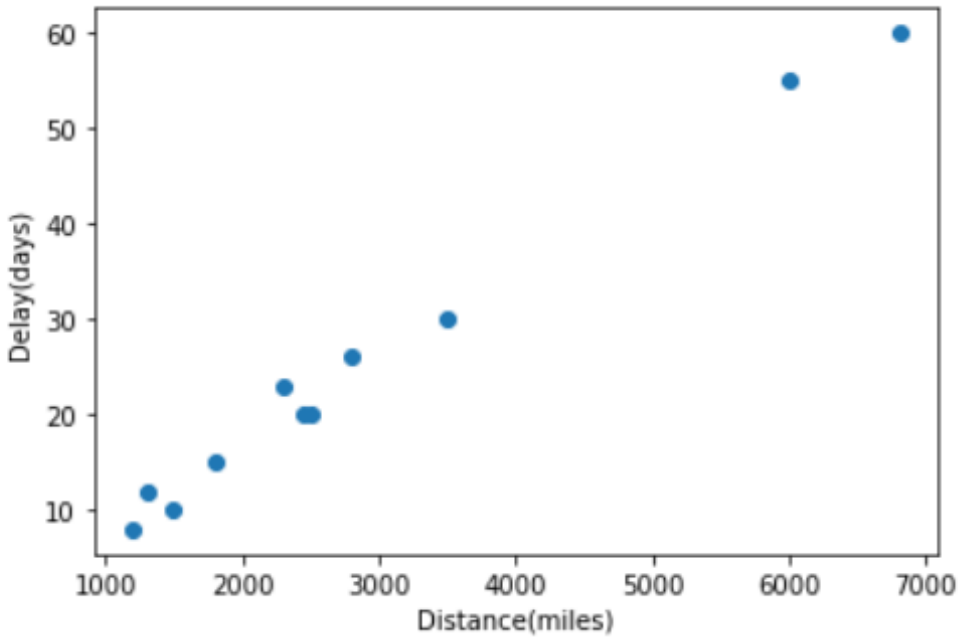

Figure 11. Plot of stored values before prediction

- We train our model on the data we collected and stored from the events in our experiment (we added this code also to our python file).

```
In [30]: df = pd.DataFrame(uspsDistance,columns =['Distance'])
         df["Time"]=time
         print(df)
```

```
    Distance  Time
0       6800    60
1       2500    20
2       1500    10
3       2300    23
4       1300    12
5       1200     8
6       1800    15
7       2800    26
8       3500    30
9       2450    20
10      6000    55
```

Figure 12. Data used for prediction

```
In [25]: reg.predict([[3300]])
Out[25]: array([28.85434906])
```

Figure 13. Model prediction for route distance of 3000 miles

- As expected we were able to fit the model on our data and make future predictions.

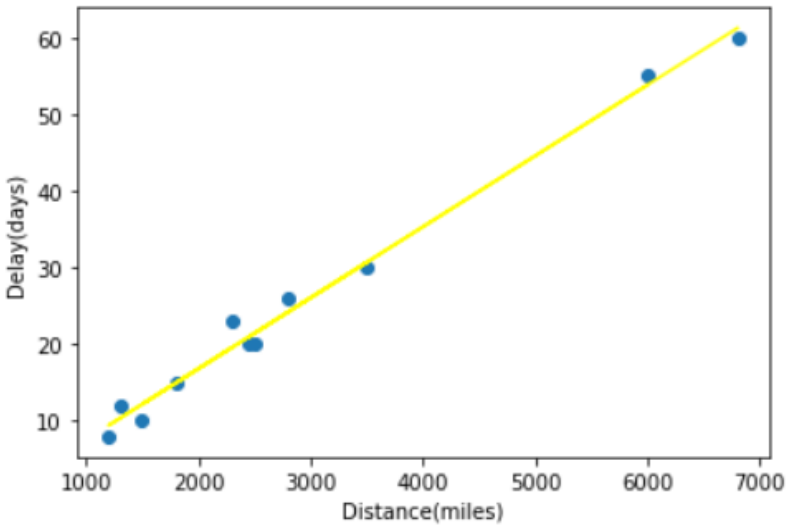

Figure 14. Plot of stored values after prediction

8. Conclusion

The main purpose of our project was to be able to establish and integrate communications between different modules and softwares as independent solutions do exist but we did not know if they could work together, which was the focus of this paper and our research work. Looking at our experiments, its safe to conclude that our driver program can communicate with blockchain and perform advanced operations like calling exposed APIs (example usps) and also running complex features like performing regression on the data in blockchain during run time.

9. Limitations and Challenges

- Challenges include needing more of the code implementation as opposed to system design in order to test functions for users to be able to purchase cannabis within the metaverse using digital currency and smart contracts.
- As a team, we had such a short amount of time to complete the project,that we had to considerably scale down the project complexity to be able to have a working prototype by the end of the term.
- It became quite a challenge to learn Solidity without a related background in Blockchain.
- Certain features we attempted to implement belonged to experimental modules in Solidity called 'pragma experimental ABIEncoderV2', and are not recommended for production.
- Tracking events using Python was a challenge as only a limited set of functions worked for testing. Web3 limited our team significantly due to limitied availablity of resources online
- Ultimately ganache was causing us more problems than the original due to our project being on windows operating system.

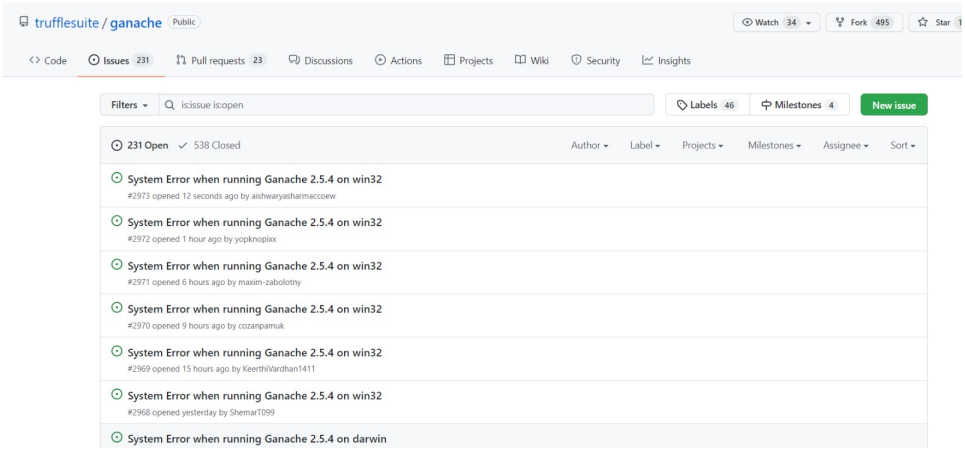

Figure 15. Ganache Issue raised by us

10. Future Work

Next steps would consist of implementing and testing functions for users to purchase cannabis, within the metaverse, using digital currency and smart contacts. Handling Real-time data from

the API responses to track timestamps, throughout a delivery. We want to implement AI that will be able to forecast deliveries, while providing real-time milestone markers as this would be a research problem for us to explore further. We also plan to look at failed use cases and security aspects of our system in order to handle as many vulnerabilities possible within the code. This project consists of multiple small problems clubbed into one to form a complex system, which during integration would be a huge challenge.

REFERENCES

[1] Retail spatial evolution: paving the way from traditional to metaverse retailing

[2] AI-Powered Blockchain - A Decentralized Secure Multiparty Computation Protocol for IoV

[3] Automating Mid- and Long-Range Scheduling for the NASA Deep Space Network Mark D. Johnston* and Daniel Tran* *Jet Propulsion Laboratory, California Institute of Technology 4800 Oak Grove Drive, Pasadena CA USA 91109

[4] A Survey of Transport Protocols for Deep Space Communication Networks International Journal of Computer Applications (0975 – 8887) Volume 31– No.8, October 2011

[5] H. Balakrishnan, V. N. Padmanabhan and R. H. Katz,"The Effects of Asymmetry on TCP Performance," Proc. ACM MOBICOM Hungary, pp. 77-89, September 1997.

[6] M.Allman, D.Glover, and L.Sanchez,"Enhancing TCP over satellite channels using standard mechanisms," IETF, RFC 2488, January 1999.

[7] S.Burleigh, A.Hooke, et al., "Delay-Tolerant Networking: An Approach to Interplanetary Internet," IEEE Communications Magazine, Vol.41, Issue 6, pp. 128-136, June 2003.

[8] O. B. Akan, J.Fang and I. F. Akyildiz,"TP-Planet: A Reliable Transport Protocol for InterPlaNetary Internet", IEEE/SAC, Vol. 22, No. 2, pp 348-61, February 2004.

[9] M. Young, The Technical Writers Handbook. Mill Valley, CA: University Science, 1989.

[10] Wang S., Qu X. (2019) Blockchain Applications in Shipping, Transportation, Logistics, and Supply Chain. In: Qu X., Zhen L., Howlett R., Jain L. (eds) Smart Transportation Systems 2019. Smart Innovation, Systems and Technologies, vol 149. Springer, Singapore. https://doi.org/10.1007/978-981-13-8683-1_23

[11] Mittal, Akash."Truffle Suite Tutorial: How to Develop Ethereum Smart Contracts - LogRocket Blog." LogRocket Blog, 2 Nov. 2021, https://blog.logrocket.com/truffle-suite-tutorial-develop-ethereum-smart-contracts/.

[12] Douglas, Joshua."Introduction to Smart Contracts — Ethereum.Org." Ethereum.Org, 10 Jan. 2022, https://ethereum.org/en/developers/docs/smart-contracts/.

[13] Newton, Casey (2021-07-22). "Mark Zuckerberg is betting Facebook's future on the metaverse". The Verge. Archived from the original on 2021-10-25. Retrieved 2021-10-25.

[14] Robertson, Adi (2021-10-04). “What is the metaverse, and do I have to care?”. The Verge. Retrieved 2022-03-09.

[15] Clark, Peter Allen (15 November 2021). “hat Is the Metaverse and Why Should I Care?”. Time. Retrieved 2021-12-29.

[16] O’Brian, Matt; Chan, Kelvin (28 October 2021). “EXPLAINER: What is the metaverse and how will it work?”. ABC News. Associated Press. Archived from the original on 4 December 2021. Retrieved 4 December 2021.

[17] Orland, Kyle (2021-11-07). “So what is “the metaverse,” exactly?”. Ars Technica. Archived from the original on 2021-11-09. Retrieved 2021-11-09.

[18] Brown, Dalvin (2021-08-30). “What is the ’metaverse’? Facebook says it’s the future of the Internet”. Washington Post. Retrieved 2021-11-01.

[19] Antin, Doug (2020-05-05). “The Technology of the Metaverse, It’s Not Just VR”. The Startup. Archived from the original on 2021-10-25. Retrieved 2021-10-25.

[20] Neiger, Chris. “Virtual reality is too expensive for most people — but that’s about to change”. Business Insider. Archived from the original on 2021-10-25. Retrieved 2021-10-25.

[21] Rhys Wood (2021-10-14). “his Oculus VR headset could feature lifelike resolution – here’s why that matters”. TechRadar. Retrieved 2022-01-09.

[22] Lewis, Leo; Davies, Christian; Jung-a, Song (2022-01-05). “Investors gear up for ’gold rush’ in metaverse hardware”. Financial Times. Retrieved 2022-01-08.

[23] D’Anastasio, Cecilia.“The Metaverse Is Simply Big Tech, but Bigger”. Wired. ISSN 1059-1028. Retrieved 2022-01-08.

APPENDIX

Request and response for usps apis which is now a part of future scope

I. USPS PACKAGE TRACK API

2.2.1 Sample Request

Request: Package Track

```
<TrackRequest USERID="XXXXXXXXXXX">
<TrackID ID="XXXXXXXXXX1"></TrackID>
<TrackID ID="XXXXXXXXXX2"></TrackID>
</TrackRequest>
```

2.3.1 Sample Response

Response: Package Track

```
<TrackResponse>
<TrackInfo ID="XXXXXXXXXX1">
<TrackSummary> Your item was delivered at 6:50 am on February 6 in BARTOW FL 33830.</TrackSummary>
<TrackDetail>February 6 6:49 am NOTICE LEFT BARTOW FL 33830</TrackDetail>
<TrackDetail>February 6 6:48 am ARRIVAL AT UNIT BARTOW FL 33830</TrackDetail>
<TrackDetail>February 6 3:49 am ARRIVAL AT UNIT LAKELAND FL 33805</TrackDetail>
<TrackDetail>February 5 7:28 pm ENROUTE 33699</TrackDetail>
<TrackDetail>February 5 7:18 pm ACCEPT OR PICKUP 33699</TrackDetail>
</TrackInfo>
<TrackInfo ID="XXXXXXXXXX2">
<TrackSummary There is no record of that mail item. If it was mailed recently, It may not yet be tracked. Please try again later. </TrackSummary>
</TrackResponse>
```

Figure 16. Sample Tracking request response

II. USPS PACKAGE TRACK API

5.2.1 Sample Request

Request: PTSPod

```
<PTSPodRequest USERID="XXXXXXXXXXX">
<TrackId>XXXXXXXXXX</TrackId >
<ClientIp>127.0.0.1</ClientIp>
<SourceId>XXXXXX</SourceId>
<MpSuffix>9402</MpSuffix>
<MpDate>2009-07-02 00:42:23.35744</MpDate>
<RequestType>Email</RequestType>
<FirstName>John</FirstName>
<LastName>Doe</LastName>
<Email1>test@email.com </Email1>
<Email2></Email2>
<Email3></Email3>
<TableCode>T</TableCode>
<CustRegID>1234567890</CustRegID>
</PTSPodRequest>
```

5.3.1 Sample Response

Response: PTSPOD

```
<PTSPODRESULT>
<ResultText>Your Proof of Delivery record is complete and will be processed shortly.</ResultText>
<ReturnCode>0</ReturnCode>
</PTSPODRESULT>
```

Figure 17. Sample Proof request response