


Peer-to-Peer Optimal Solar Energy Trading using Proof-of-Authority Blockchain

Mohammad Rafaquat Alam, Teoh Jing Yang

Abstract—With Malaysia’s growing population and improved lifestyle, energy demand will only increase posing new challenges and problems to the energy sector. We propose a peer-to-peer web-based solar energy trading web application with the aim of optimizing energy usage by homes, such that those with higher requirements, can trade energy instead of paying higher bills. Thus also providing a second income stream for prosumers for their production of the tradable renewable solar energy. We use Machine Learning to predict energy consumption and production. Based on these predictions, users can buy and sell renewable energy. To ensure that maximum social welfare, we implement a Double Auction Mechanism (DAM) to optimally allocate tradable energy between prosumers. This transactions are stored on a Proof-of-Authority Distributed Ledger. The back end is built using Go programming language due to its speed, easy syntax and its increasing popularity as a server side language. The Angular framework is used for the front end development with Typescript used for middleware service implementation. Energy Forecasting and Double Auction Mechanism are written using Python libraries. Moreover, the platform uses the MongoDB NoSQL cloud database to securely and reliably store data. The Locust stress testing tool is used for performance benchmarking.

Index Terms—Solar Energy, Energy Forecasting, Double Auction, Blockchain, Frontend and Backend. 

I. INTRODUCTION

AS access to electricity increases with rising number of appliances and electrical devices in households, it is only fair that the energy consumption for the average person throughout history has been increasing steadily [1], [2]. Grids in Malaysia are controlled by Tenaga Nasional Berhad (TNB), with electricity flowing through main transmission lines from the generators all the way to distribution lines which in turn is forwarded to residential areas. Without a doubt, as the energy demand and consumption keeps increasing, the transmission lines will have to be upgraded eventually [3]. This will cost copious amounts of time and money. If nothing is done the transmission lines will become extremely congested and unable to fulfill energy demands. Another downfall of the energy sector in some developed countries is that houses with solar panels, do not utilize the generated surplus energy to its full potential. They do not benefit in any way from the surplus energy they generate as all of it is dissipated to the grid for free [4]. This reduces morale for the general public as there is no incentive to produce the extra energy. This demotivates them from renewable energy generation and usage.

We combat the aforementioned problems with our P2P Energy Trading web application that guarantees prosumer rewards by trading their produced solar energy. We use Energy Forecasting algorithms to predict energy consumption and

production of users. This is used to calculate the Optimal Energy that can be allocated to the prosumer (buyer) from other prosumers (bidders) using the Double Auction Mechanism (DAM). The trading energy and fiat amount are stored as part of transactions in a block in a Proof-Of-Authority Distributed Ledger. So buyers can order energy on the market and bidders can trade their produced energy in exchange for fiat currency. A portion of this fiat amount is also forwarded to TNB since the system relies on their grid network. This way electricity is not pulled from the mains resulting the overall demand and stress on the transmission lines to be reduced [4], [5]. Consequently, massive infrastructure upgrade costs and resources can be avoided. Furthermore, an incentive is created for people to own solar panels and start trading on the P2P platform, thus increasing the usage of renewable energy leading to a cleaner and greener environment.

A. Organization of Paper

Rest of the paper is organized as follows. Section II elaborates the methods used to implement the solution and the justification behind them. Detailed description of the results from each method are illustrated in Section III. Section IV discusses the issues in the system and the conclusion is provided in Section V. The appendix section describes the meaning of the equation variables from the Double Auction Mechanism.

II. METHODS

The workflow of the application is that first Energy Forecasting is done to allow users to make orders and bids. Then DAM is run for Optimal Energy Allocation and finally the transaction is stored in the blockchain. This is discussed in detail in the sections below.

A. Energy Forecasting

1) *Why Energy Forecasting:* For our implemented web application, we are assuming that the solar energy a prosumer generates is stored in a battery and the energy balance is stored in a smart metre in their homes. Using the stream of energy data from the smart metre, we forecast the amount of energy consumed as well as produced for a given day. By predicting the energy consumption, we can limit the prosumer (buyer) to a max energy value that they can order. This ensures that extremely large energy amounts, that even the grid cannot produce, are never ordered. Predicting the energy production limits the prosumer (bidder) to a max tradable energy value

they can use to make a bid on an energy order. This ensures, that no prosumer makes a bid that exceeds the max amount they are expected to produce.

2) *How Energy Forecasting is done:* The energy forecasting uses the Machine Learning technique of Triple Exponential Smoothing [10] with Additive Trend and Seasonality where the latest two data points are used for seasonal period [9]. The training also includes Trend Damping to prevent the rise of unrealistic trends and account for Seasonal Irregularity [10]. Once training is done using all the data points upto the current time of the day, a prediction is generated for energy consumption/production for the next 30 minutes from the current time. The prediction is the maximum energy forecast for consumption or production depending on what feature the prosumer is making use of. If the prosumer wants to make an order, they are shown their forecast graph for energy consumption as in Figure 1. If the prosumer wants to make a bid on an energy request, they are shown their forecast graph for energy they can produce as shown in Figure 2. In terms of implementation, we used the Python Pandas for data processing and Statmodels for its ExponentialSmoothing library.

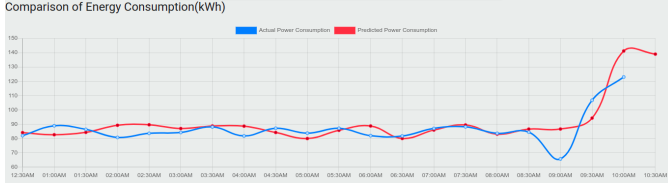


Fig. 1. Prosumer (buyer) energy consumption forecast for the next 30 minutes

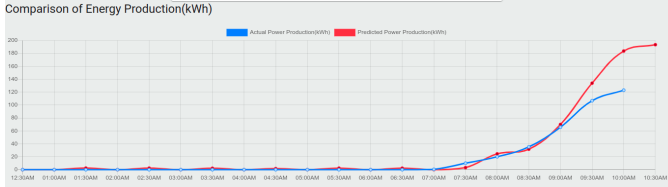


Fig. 2. Prosumer (bidder) energy production forecast for the next 30 minutes

B. Optimal Energy Allocation

1) *Social Welfare Maximisation (SWM):* The SWM problem is the objective function that we hope to maximise to fulfill the energy request of prosumers by also maximising the energy supply from bidders whilst considering the cost to generate the energy. The amount of energy that would satisfy a prosumer (buyer) based on their smart metre data and energy consumption forecast is defined by the satisfaction function in [6] as:

$$U_i(E_i^n) = w_i \left[\ln \left(n \sum_{j=1}^J (e_{ij}^n - e_i^{n,min}) + 1 \right) \right] \quad (1)$$

See the appendix section for more details. Next, we need to consider the cost incurred for the bidding prosumer when they trade the produced energy as per the production forecast. The cost function is given in [6] as:

$$L_i(S_j^n) = c_1 \sum_{i=1}^I (s_j^n)^2 + c_2 \sum_{i=1}^I (s_{ji}^n) \quad (2)$$

See the appendix section for more details. Now that we know how much energy would satisfy the prosumer as shown in Eq. (1) and the cost the bidder has to bear to produce and trade energy as in Eq. (2), we can determine the objective function for SWM as:

$$SWM : \max_{E^n, S^n} \sum_{i=1}^I U_i(E_i^n) - \sum_{j=1}^J L_j(S_j^n) \quad (3)$$

The SWM is subject to the following constraints.

$$\begin{aligned} \text{Subject to : } e_i^{n,min} &\leq \eta \sum_{j=1}^J e_{ij}^n \leq e_i^{n,max}, \forall i \in E, \\ \sum_{i=1}^I s_{ji}^n &\leq s_j^{n,max}, \forall j \in Z, \\ \rho s_{ji}^n &= e_{ij}^n, \forall i \in E, \forall j \in Z, \\ e_{ij}^n &\geq 0, \forall i \in E, \forall j \in Z. \end{aligned} \quad (4)$$

2) *Optimal Allocation Problem (OAP):* The OAP, ensures no one prosumer can out bid everyone else, meaning all bidders are guaranteed to receive fiat reward as long as they bid on the order/request. Moreover TNB, as the grid providers, are also guaranteed to receive a reward since the system is designed to use their infrastructure to distribute the energy. The OAP is defined in [6] as :

$$OAP : \max_{E^n, S^n} \sum_{i=1}^I \sum_{j=1}^J [b_{ij}^n \ln e_{ij}^n - p_{ji}^n s_{ji}^n] \quad (5)$$

See Appendix for details. Both ‘OAP’ and ‘SWM’ have same subject to constraints as described in Eq. (4). Hence we carry out constraint relaxation through Lagrangian method [6] to find the local maximum [10]. So, energy receivable by a buyer, e_{ij}^n and energy tradable by a bidder, s_{ji}^n , after applying Lagrangian method is:

$$e_{ij}^n = \frac{b_{ij}^n [(n \sum_{j=1}^J e_{ij}^n - e_i^{n,min}) + 1]}{nw_i} \quad (6)$$

where $e_i^{n,min}$ is the fixed minimum energy for a buyer, and:

$$s_{ji}^n = 2c_1 s_{ji}^n + c_2 \quad (7)$$

where, c_1 and c_2 are cost factors for the bidder.

3) *Double Auction Mechanism (DAM):* The Double Auction Mechanism (DAM) solves the OAP via Lagrange method. As shown in Figure. 3, the application runs the DAM every 30 minutes taking all closed requests, looping through them and applying Eq. (6). and Eq. (7). on each bid. The Langrange scales down the bid energy amount by 10 to 15% on average to prevent one bidder to dominate. So, once all the bidders are considered, the system checks whether the total energy demand made by the prosumer in the order is satisfied or not. If demand is satisfied, the remaining fiat amount that is not received by any bidder is transferred to TNB. If demand is not satisfied the surplus energy is taken from TNB. The rate of energy is capped at RM 0.20 per kWH $\pm 20\%$ for all bidders as well as TNB. So, all bidders and TNB receive fiat currency as reward and buyer only pays for the amount of energy they need which is always fulfilled.

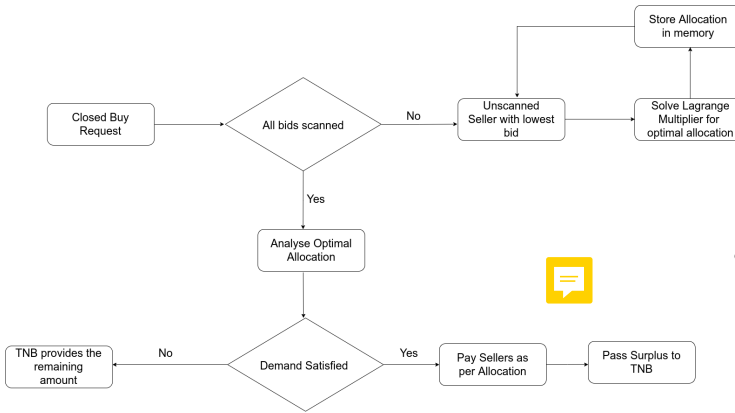


Fig. 3 Flowchart for double auction with demand and supply response

C. Blockchain

As shown in Fig. 4, the web application uses a Proof of Authority (PoA) consensus mechanism for the blockchain.

1) *Roles in the Blockchain:* Only validators are given the authority to mine new blocks and add them to the blockchain. We also have a new kind of users called clerks who are there for validator security and accountability checks. This is to ensure that validators do not conspire to work against public interest. If validator accounts are compromised, then clerks can help recover the system.

- *Validator:* A validator can verify transactions, mine new blocks, add and discard blocks. Since we are using a Proof of Authority consensus mechanism, the validators undergo a rigorous registration process where they need to reveal their identities. Their reputation is at stake which means if they go against the interest of the prosumers, then their status as validators will be revoked and made known to the greater community.
- *Clerk:* Clerks provide an additional layer of integrity check on the blockchain to ensure that validators do not conspire against the community. After the addition of every new block, they receive an updated local copy of the blockchain and user accounts. They use their local blockchain copy for doing integrity check on the central one. Unlike validators, any normal node can be made a clerk and they do not need to be rigorously identified. Clerks also provide the benefit of recovering the blockchain in case a validator account is compromised since they have a 51% agreement on the valid blockchain.

2) Blockchain pipeline:

- *Signing Smart Contracts:* After the DAM is run every 30 minutes, the new pool of transactions are broadcasted over the network to all the validators. The validator who receives the transaction pool first, will verify each transaction where they check whether the buyer has sufficient balance or not. If so, then that transaction is marked as verified and made part of a temporary block. If a buyer does not have the required balance then the transaction is marked invalid. Once all the transactions have been checked, and added to the temporary block, it is then broadcasted to all the remaining validators. These

validators use their local copy of user accounts to verify each transaction in the temporary block. They then use the nonce of the temporary block to hash the transactions. If this hash matches with that of the temporary block for all validators, then the temporary one is made permanent and added to the central blockchain. The last validator who checks the hash finalizes/signs the smart contract.

- *Discarding Blocks:* If there is a validator who does not find a match for the hash, then their local copies of user accounts and blockchain is updated. Then the check is done again. If the hash fails to match a second time, then that block is discarded permanently. The non-match signifies that a transaction was manipulated in the central blockchain so the discarding is justified.
- *Integrity Check:* After the formation of 5 new blocks, an integrity check is triggered where the clerks verify each transaction in the latest permanent block in the central blockchain. They use their local copy of user accounts to verify each transaction in the latest block. Then use the nonce of the latest block from central blockchain to hash the transactions from the latest block in their local blockchain. If this hash matches that in the central blockchain for more than 50 percent of the clerks, then there is no issue but if the match is less than 50 percent, then an integrity check is issued. This goes through the local blockchain copy of each validator and compares the hash of the latest block. The validator(s) whose hash has a mismatch is then flagged since they may have tempered the central blockchain. In this case the validator access may be revoked thus safeguarding public interest.

III. RESULTS

We now discuss the results that we got from trying to fulfill the project promises in terms of social welfare by generating income, suitability of the application design and performance of the application and blockchain.

A. Energy Trading As A Source Of Income

As mentioned earlier, the DAM of the web application ensures that all bidders can make some fiat income as long as they make a bid. This serves as an incentive for more people to invest in equipment to generate electricity using solar panels. The income chart is accessible from the user profile page as shown in Fig. 5. The chart shows the total income and energy traded for the days on which the user made a bid offer.

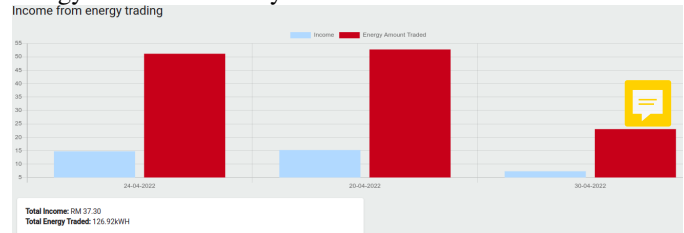


Fig. 5. User income from bidding on the application

Since we are using the grid infrastructure provided by TNB, the application ensures that they receive some fiat reward for their participation in the energy

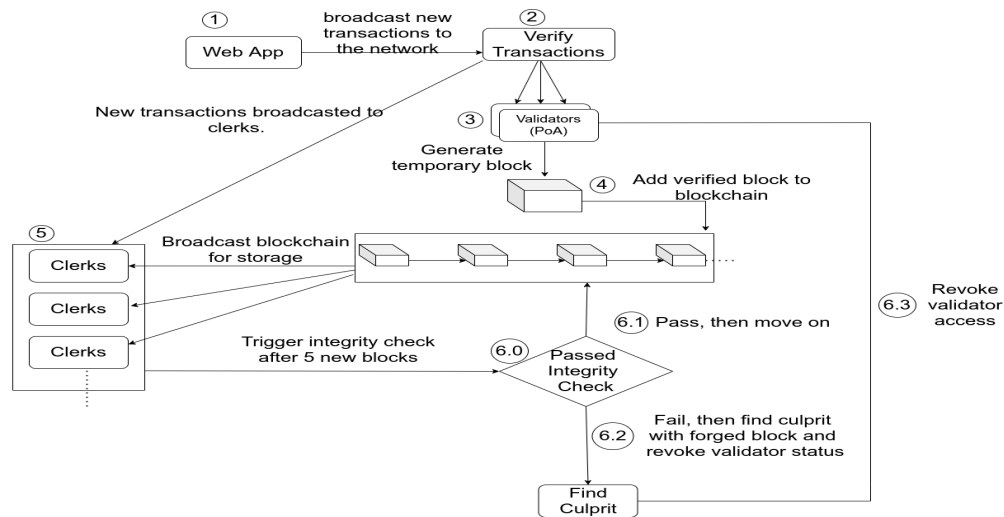


Fig. 4. Proof of Authority blockchain with accountability check via clerks

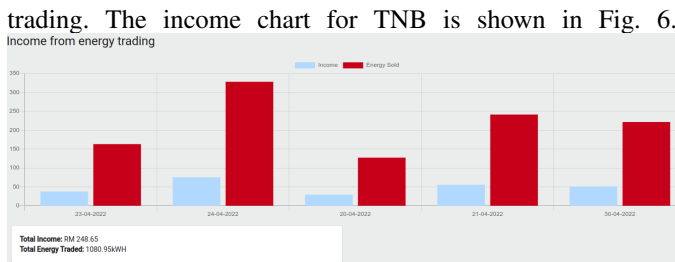


Fig. 6. Potential income for TNB

The TNB account is a validator account. No matter what bids are made on an energy request, TNB always provides a fraction of the energy demand and hence this income is always guaranteed. If there are no bids on an energy request, then the full demand is fulfilled by TNB. In such a case they receive the full price of the energy.

B. Blockchain Management

The blockchain exists in two formats. One of them is the centralised one, stored in the database and the other is a local copy stored in the validators' and clerks' device as a .JSON file.

1) *Blockchain on the frontend of web application:* As shown in Fig. 7, the central blockchain is available to all users from the "blockchain" page of the web application. The index, hash, nonce and the previous hash is shown for each block.

Index	Hash	Previous Hash	Nonce
0	0036681436deefad74e7cf50961abc81858e15e31967c287da522724e33		6
1	08f0cbac423e303e50f03f84f5ba1ee2d5d0c1dc4734355fa7cd55daa1353a5	0036681436deefad74e7cf50961abc81858e15e31967c287da522724e33	5
2	05b4d60f384a4925aa9a51c1ff4d3b4316afa6d544e6b47795892b59313615862	08f0cbac423e303e50f03f84f5ba1ee2d5d0c1dc4734355fa7cd55daa1353a5	9
3	0b0ba7519fc3208cf682f68b3f1db26a01891b27ce66e1c6e2d9afce383a022	05b4d60f384a4925aa9a51c1ff4d3b4316afa6d544e6b47795892b59313615862	1
4	042c6dc5f63766c517ca006fec1ae1b2ebba77604658623b2947a478c0b2e1	0b0ba7519fc3208cf682f68b3f1db26a01891b27ce66e1c6e2d9afce383a022	15

Fig. 7. Index, Hash, Block Info Icon, Previous Hash and Nonce are displayed on the frontend

In addition to these, when the info icon is clicked, users can see some general information regarding the transactions in each

block: the buyer, total energy and fiat traded and the income TNB generates from this transaction. This is shown in Fig. 8.

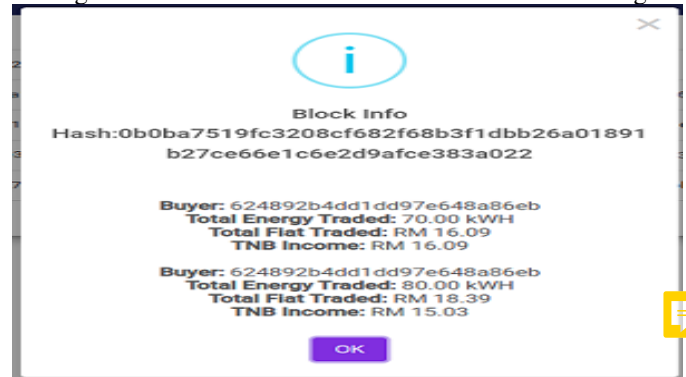


Fig. 8. Transaction information in the block

2) *Blockchain on the backend of web application:*

As mentioned before, the backend stores the blockchain in a central database as shown in Fig. 9. This is used as the primary blockchain throughout the system. The frontend uses the data available in this blockchain.

```

{
  "_id": ObjectId("6264cea2ff045b239c95a663"),
  "index": 1,
  "data": {
    "hash": "08f0cbac423e303e50f03f84f5ba1ee2d5d0c1dc4734355fa7cd55daa1353a5",
    "prevhash": "0036681436deefad74e7cf50961abc81858e15e31967c287da522724e33",
    "nonce": "5"
  }
}

```

Fig. 9. Blockchain as stored centrally using mongodb

The other blockchain format is a local copy stored in the validators' and clerks' device as a .JSON file and this is shown in Fig. 10. As discussed before, the application uses this local copy to verify transactions and check the hash in the central database before finalizing the blocks. The app always updates the local copy using the central one when the verification and

integrity checks are done. This means there is no chance for clerks and validators to edit the local copy and manipulate the central blockchain.

```

{
  "index": 2,
  "data": {
    {
      "BuyerId": "623722a33811066ba71aed41",
      "BuyerPayable": 34.480434782608704,
      "BuyerEnReceivableFromAuction": 22.739543331907836,
      "BuyerEnReceivableFromTNB": 127.26045668809216,
      "AuctionBids": [
        {
          "SellerId": "6235ead63fc9762da8b7e54",
          "OptEnFromSeller": 22.739543331907836,
          "OptSellerReceivable": 5.2271289389476845,
          "SellerFlatBalance": 20000,
          "SellerEnergyBalance": 800
        }
      ],
      "TNBReceivable": 29.253305843601018,
      "Verified": true,
      "Tid": "6264d05456771e5a219a9a56",
      "Checks": 0,
      "Date": ""
    }
  ],
  "hash": "05b4d60f384a4925aa9a51cfff4d3b4316afa6d544e6b47795892b59313615862",
  "prevHash": "08f0cbac423e303e50f03f84f5ba1ee2d5d0c1dcf4734355fa7cd55daa1353a5",
  "nonce": "9"
}

```

Fig. 10. Snippet of a block stored in the local blockchain file

C. Web Application Testing

1) *Load Testing*: When the platform is eventually released to the public, it has to serve lots of users concurrently without any issues. We used a swarming tool called Locust to simulate such scenarios. The load testing simulates 100 normal users concurrently using the system. Each user first visits the homepage, creates an order, makes a bid, visits the marketplace and finally the blockchain page. Such sequence ensures real user behaviour when the system is deployed. Fig. 11 and Fig.12 respectively show the system performance when handling concurrent user requests and system responses. From both graphs, we can see that there are no failures meaning the system works as intended under normal loads.

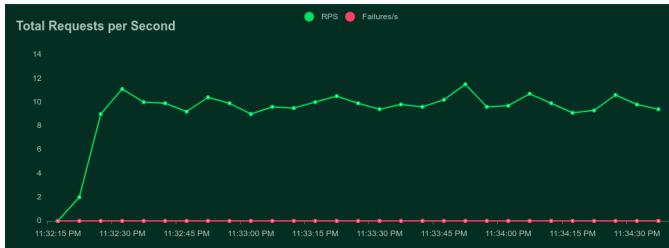


Fig. 11. Handling concurrent requests on backend servers.

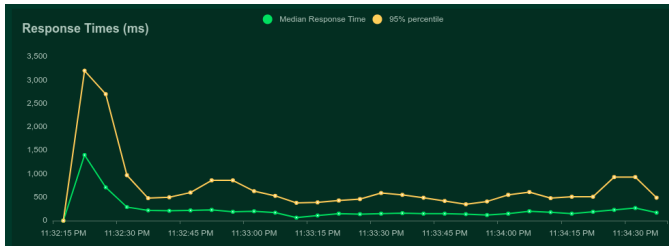


Fig. 12. System responding to concurrent user requests.

2) *Stress Testing*: Stress testing is done to find out at what capacity the system fails. This will help us determine which upgraded database version to use during deployment since the current one is using the free cluster service from mongodb. It also shows us how gracefully the system fails. The results from stress testing are shown in Fig. 13 and Fig. 14 respectively. In both cases the request failures and response time started to increase when we hit the 1000 user threshold.

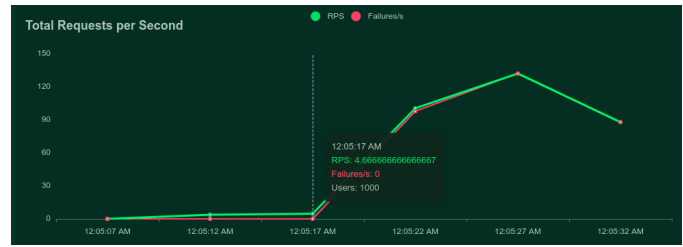


Fig. 13. Request handling by backend servers under high stress.

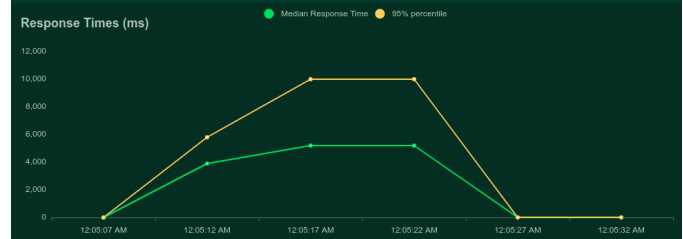


Fig. 14. Response handling by backend servers under high stress.

3) *Blockchain Performance*: As the web application is still in prototype phase, the mining of blocks were kept to the lowest difficulty, that is each block is solved/mined when there is nonce that returns one zero at the very start of the hash string. With this level of difficulty, we tested the blockchain mining time with increasing transaction size as shown in Fig. 15. Since the hashing/mining algorithm only increments the nonce and checks for zero character at the start of the hash string, there is no specific observation. However, in the real world scenario, as block size increases with increasing transaction sizes, the difficulty would need to be increased and hence the mining time will increase as well.

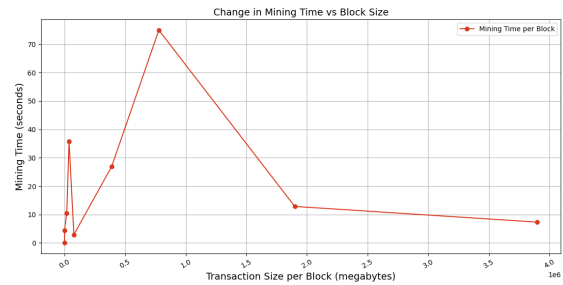


Fig. 15. Time taken to mine a block with increasing block size.

D. Source Code

The web application over a period of 6 months from December 2021 to May 2022. The code is made available at: https://gitlab.com/mohammad_rafaquat_alam/imdc_p2p_energy_trading If there are issues in accessing the repository please kindly email: md.rafaquatalam98@gmail.com. A demonstration video is made available at this link to showcase the usage of the web application.

IV. DISCUSSION

A. Security Concerns

The key security issue of the system is that the private and public keys of all the users are hashed and stored in the

database. Even though the hashing algorithm is not reversible this is still not good software development practice. Attackers can launch a dictionary attack with a known hash table and therefore obtain the real private key. So, further work needs to be done to write a mechanism to store and extract the private key of users from the user devices as opposed to the database.

B. Database Upgrade

As shown in the results section, the free mongodb database clusters that are used are not suitable for large scale use. The system crashes when 1000 users are using the system concurrently. To fix, this funding is required to upgrade to a better and reliable mongodb service. The underlying database code would remain the same so this fix is very simple.

V. CONCLUSION

Based on all the previous sections, it is clear that the Blockchain technology is here to stay and keep penetrating even more industries. P2P Optimal Solar Energy trading has a huge potential to make ripples through the energy industry. In conclusion, team MY157 have successfully implemented a peer to peer optimal solar energy trading platform using blockchain technology.

APPENDIX

A. Optimal Energy Allocation Equations

$$U_i(E_i^n) = w_i[\ln(n \sum_{j=1}^J (e_{ij}^n - e_i^{n,min}) + 1)]$$

where E^n is the satisfactory energy receivable for prosumer, w_i is the battery charging willingness, n is the battery charging efficiency, J is the total number of bids on the request, e_{ij}^n is the energy the prosumer(buyer) can receive from the bidder as per the prosumer's consumption forecast. Next we need to consider the cost incurred for the bidding prosumer when they trade the energy they produced as per the production forecast. The cost function is given in [6] as:

$$L_i(S_j^n) = c_1 \sum_{i=1}^I (s_j^n)^2 + c_2 \sum_{i=1}^I (s_{ji}^n)$$

where S^n is the energy that bidder can provide, I is the request made by the prosumer, c_1 and c_2 are cost factors for the bidder, s_j^n is the energy the bidder can produce as per their forecast, s_{ji}^n is the energy the bidder wants to trade with the buyer from the total produced. Now that we know how much energy would satisfy the prosumer as shown in Eq. (1) and the cost the bidder has to bear to produce and trade energy as in Eq. (2), we can determine the objective function for SWM as:

$$SWM : \max_{E^n, S^n} \sum_{i=1}^I U_i(E_i^n) - \sum_{j=1}^J L_j(S_j^n)$$

$$\text{Subject to : } e_i^{n,min} \leq \eta \sum_{j=1}^J e_{ij}^n \leq e_i^{n,max}, \forall i \in E,$$

$$\sum_{i=1}^I s_{ji}^n \leq S_j^{n,max} \forall j \in Z,$$

$$\rho s_{ji}^n = e_{ij}^n, \forall i \in E, \forall j \in Z,$$

$$e_{ij}^n \geq 0, \forall i \in E, \forall j \in Z.$$

B. Optimal Allocation Problem Equations

$$OAP : \max_{E^n, S^n} \sum_{i=1}^I \sum_{j=1}^J [b_{ij}^n \ln e_{ij}^n - p_{ji}^n s_{ji}^n]$$

where b_{ij}^n is the price the bidder wants from the prosumer, p_{ji}^n is the price the prosumer is willing to pay to bidder,

$$e_{ij}^n = \frac{b_{ij}^n [(n \sum_{j=1}^J e_{ij}^n - e_i^{n,min}) + 1]}{nw_i}$$

where $e_i^{n,min}$ is the fixed minimum energy for a buyer, and:

$$s_{ji}^n = 2c_1 s_{ji}^n + c_2$$

where, c_1 and c_2 are cost factors for the bidder.

This piece of work would have not been possible without the support of our family, friends and university professors and their firm belief in my potential. The team is immensely grateful for the efforts of my supervisor Dr. Tan Wen Shan, for believing in our capabilities since he chose us for this project, and for supporting and trusting our decision making, until the very end of it.

REFERENCES

- [1] G. "U.S. energy facts explained - consumption and production - U.S. Energy Information Administration (EIA)", Eia.gov, 2021.
- [2] H. Ritchie and M. Roser, "Energy", Our World in Data, 2021.
- [3] "Outline of Malaysia's electricity system - Power Engineering International", Power Engineering International, 2021.
- [4] J. Yang, A. Paudel, H. B. Gooi, and H. D. Nguyen, "A proof-of-stake public blockchain based pricing scheme for peer-to-peer energy trading," Applied Energy, vol. 298, p. 117154, 2021.
- [5] S. Aggarwal and N. Kumar, "A consortium blockchain-based energy trading for demand response management in vehicle-to-grid," IEEE Transactions on Vehicular Technology, vol. 70, no. 9, pp. 9480-9494, 2021.
- [6] The consortium paper
- [7] The demand and supply one by Dr. Tan
- [8] Proof of Authority
- [9] Time-series Forecasting - Complete Tutorial (indian guy)
- [10] A Gentle Introduction to Exponential Smoothing for Time Series Forecasting in Python
- [11] A Gentle Introduction To Method Of Lagrange Multipliers