ETHERSTWEB – An Ethereum-Based Distributed Application for User Information Trustworthiness Verification

Chong-Gee Koa[1], Swee-Huay Heng[2*], Ji-Jian Chin[3]

[1,2]Faculty of Information Science and Technology, Multimedia University, 75450, Melaka, Malaysia; E-mail: shheng@mmu.edu.my

[3]School of Engineering, Computing and Mathematics (Faculty of Science and Engineering), University of Plymouth, Drake Circus, Plymouth PL4 8AA, United Kingdom

Abstracts: Fake news and misinformation are prevalent on the Internet in this day and age. The popularity of social media such as Facebook, Instagram, TikTok has encouraged the people to share news or information over Internet without knowing whether it is a truth or fake information. The outbreak of COVID-19 affected the social activities of all levels of society as people were asked to stay at home to self-quarantine. It indirectly encourages the growth of the social activities online. Even though COVID- 19 is no longer a big threat to the world, people already used to rely on the Internet for the daily activities. Therefore, the trustworthy of the information on the Internet is getting crucial. ETHERST is a blockchain- based PKI that implemented rewarding and punishment mechanism using Ethereum ECR-20 token named PKIToken to improve the trustworthiness of information published by the blockchain nodes. In this paper, we implement an Ethereum-based distributed application (dapp) that allowed the community members to act on trusting and untrusting on information provided by any of the members. With the ETHERST framework as the backend, the ETHERSTWEB is equipped with the rewarding and punishment mechanism to keep the community from misusing the trusting and untrusting action to maintain the trustworthiness of information of a member. With the PKIToken amount level implemented in the backend, it provides an index to the trustworthiness of a member in the community. It has the advantages over the existing rating or review systems that are commonly implemented in many traditional web applications.

Keywords: Blockchain, Ethereum, ETHERST.

1. INTRODUCTION

The Internet has become saturated with misinformation and fake news. Social media platforms like Facebook, Instagram, and TikTok have contributed to this problem by encouraging people to share information without verifying its accuracy (Aldwairi & Alwahedi, 2018). The COVID-19 pandemic forced people to stay at home and practise social distancing, leading to an increase in online social activities (De', Pandey, & Pal, 2020). Although the pandemic is no longer a major threat, people have grown accustomed to relying on the Internet for daily activities. As a result, the trustworthiness of online information has become increasingly important.

ETHERST (Koa, Heng, & Chin, 2021) is a blockchain-based PKI that implements a reward and punishment mechanism. Blockchain technology is popular due to its features of immutability, trustworthiness, distributed and decentralisation, therefore, it is an excellent candidate to be implemented as a backend framework to improve the information trustworthiness by the community members of an Internet application.

In this paper, we introduce ETHERWEB, a Web 3.0 decentralised application (dapp), which is integrated with features to allow application users to verify the user information accuracy. In the next section, we look into the problem of fake data and inaccurate information about users in existing web applications, using e-commerce as an example to elaborate on how it can be a frustrating experience for users.

1.1. Organisation of the Paper

The rest of the paper is organised as follows. We discuss the online fake news and information issues in Section 2. Next, in Section 3, we review the existing research on fake news and information issues and the proposed blockchain solutions. In the subsequent Section 4, we briefly explain the blockchain-based PKI framework named ETHERST that will be used as the backend of our decentralised application presented in this paper. We outline the development stack of the ETHERSTWEB and the steps for building and deploying it to the Internet in Section 5. The features and screenshots of ETHERSTWEB are presented in Section 6. In the same section, we also briefly explain on the integration of ETHERST with ETHERSTWEB. Finally, in Section 7, we conclude the paper with a discussion on possible future work on ETHERSTWEB and a summary of the work.

2. INFORMATION ACCURACY AND TRUSTWORTHINESS IN INTERNET APPLICATIONS

Web application users are coming from all over the place. There are some web applications that allow users to interact and form a virtual community. Misinformation such as fake news or wrong information about a product can be spread through the virtual community via web application. For example, marketplace applications like e-commerce applications have a community comprising merchants and buyers. When buyers want to purchase an item from a merchant, they will review the profile of the merchant. The existing e-commerce applications normally will provide a review subsystem that allows buyers to leave comments and ratings on the transactions done between a buyer and the merchant. A good merchant normally will have a good review and rating or reputation. Buyers will base this information to evaluate the trustworthiness of the merchant. However, with only this review and rating system, a dishonest merchant can generate fake transactions and fake buyers to improve the "trustworthiness".

In order to avoid such fake actions, we first need to have a verification process for each user regardless of merchants or buyers. If the verification process needs to be carried out by the e-commerce application company, that will incur a high cost to the company. In fact, the verification processes can be distributed to community members using a different approach from the existing review and rating system.

Another example involves the spread of fake news, which can occur through blogs or virtual communities on social networking applications. When users share unverified news on their social network platforms, the individuals in their network lack a reliable means to assess the authenticity of the news and the credibility of the user sharing it. Similar to the case with e-commerce applications we discussed earlier, a verification process is essential, and implementing distributed verification processes can offer a solution to this problem.

In this paper, we recommend the integration of blockchain technology in the distribution of verification processes by community members. Blockchain technology is a technology introduced by Satoshi Nakamoto (Nakamoto, 2008), well-known for its features of decentralisation, traceability, immutability, and currency properties. We believe it is the best option to tackle the issue of the trustworthiness of the information about the users in internet applications. Consequently, ETHERST (Koa et al., 2021), a blockchain-based public key infrastructure has been used as the backend in the dapp that we would like to propose in this paper.

3. FAKE INFORMATION ONLINE AND THE BLOCKCHAIN SOLUTIONS

The fake information online problem has been studied by researchers for many years either from the aspect of sociology or information technology. For instance, Vicario et al. (2016) discussed the spreading of misinformation online. The focus of their studies was from the aspect of sociology on how users consume information pertaining to two separate narratives: scientific news and conspiracy theories. In the aspect of information technology, some blockchain solutions have been proposed since 2019. Qayyum et al. (2019) proposed a blockchain framework that allows news publisher to publish their news on Ethereum.

In the proposal, publishers need to enrol themselves on the blockchain network before publishing the news.

In 2020, TRUSTD, a proposal introduced by Jaroucheh et al. (2020), used blockchain and collective signature technology that allows community users to judge the credibility of the content published by content writers. Each content will be given a credibility score using a customised consensus algorithm in the form of a dynamic weighted-ranking evaluation score in their proposal.

ProBlock (Sengupta, Nagpal, Mehrotra, & Srivastava, 2021), another blockchain solution with a weighted majority voting model named Probit, which incorporates into blockchain through a modified version of Proof of Trust (PoT) consensus (Bahri & Girdzijauskas, 2018). In Probit voting model, users are divided into several classes and different classes of users were given a different weight in the computation of voting scores. when news is added to the blockchain, the users can cast a vote on thenews. The final vote score is calculated with statistical methods. At the end of their proposal, a comparison analysis of Probit model with a standard mean model was presented.

Dhall, Dwivedi, Pal, and Srivastava (2021) used another type of approach, by using blockchain and keyed watermarking-based framework for social media/messaging platforms to stamp the online content to mark it as genuine to curb the spreading of fake news. Besides, the framework also observes the blockchain transactions and analyses the density and checks the frequency of the forwarding of original content.

The fake news problems still attract the attention of researchers in 2022. In the research for solving the fake news problem by Wahane and Patil (2022), they pointed out that most of the current research has focused only on theoretical frameworks, methodologies and prototypes for the use of blockchain technology. There is still a wide gap between the features of current blockchain-based proposals and the requirements of organisations, companies and online platforms (Wahane & Patil, 2022).

From our observation, the current blockchain-based solutions mainly focus on fake news and content verification. Those proposals still suffered from the Sybil attack (Douceur, 2002) because the content writer can create multiple fake accounts or identities to join the voting process which affects the result of the computation. We believe that everything should start with the content writer and publisher’s trustworthiness.

4. ETHERST - THE INFORMATION TRUSTING AND UNTRUSTING FRAMEWORK

ETHERST was invented by Koa et al. (2021) as a new type of blockchain-based public key infrastructure (PKI) that implements a reward and punishment mechanism on top of the conventional distributed public key infrastructure. It is implemented on Ethereum which is a special type of blockchain proposed by Vitalik Buterin (Buterin, 2013). ETHERST uses the capability of Ethereum that allows a set of programming codes to run on top of Ethereum and the set of programming codes called "smart contract". The smart contract allows developers to build dynamic logic that can utilise the features of blockchain technology. ETHERST consists of two smart contracts:

- PKIToken smart contract;
- ETHERST smart contract.

PKIToken is an ERC-20 token (ERC-20, 2021) that is implemented by ETHERST, as a medium toreward and punish the nodes based on their good and bad actions in the ETHERST network. ETHERST smart contract is where the core logic of ETHERST blockchain-based PKI resides. ETHERST allows users to create data in the blockchain network and sign it with their private key to claim that the data or information is created and published originally by them in the community. Meanwhile, the users can also revoke the signature that they have signed on a particular data. In addition to the conventional distributed PKI such as Web of Trust (WoT) (Ryabitsev, 2014), ETHERST provides functionalities to allow users to act to "trust" and "untrust" signed data. When users joined ETHERST, they will be pre-allocated with some PKIToken, they can use them to trust and untrust the signed data of other users. There is an algorithm behind the ETHERST to increase and reduce PKIToken owned by the users based on their trust and untrust actions. Figure 1 shows the scenarios of untrusting in ETHERST to demonstrate the algorithm of untrust. Besides, in the ETHERST smart contract, there is some logic that can prevent the Sybil attack by delaying the "trust" and "untrust" actions per node with an internal algorithm as depicted in Figure 2. Every user has its own delay period *T* initialed by ETHERST. The subsequent action of "trust" or "untrust" from a user will be failed if the action is within their delay period *T*. The value of *T* will be increased if the subsequent action of "trust" or "untrust" is earlier than the delay period to prevent a dishonest user from keep trusting self-created fake users or keep untrusting the other users. With the features that come with ETHERST, we utilise it as an information trustworthiness backend framework for ETHERSTWEB that we are going to present in the following sections.

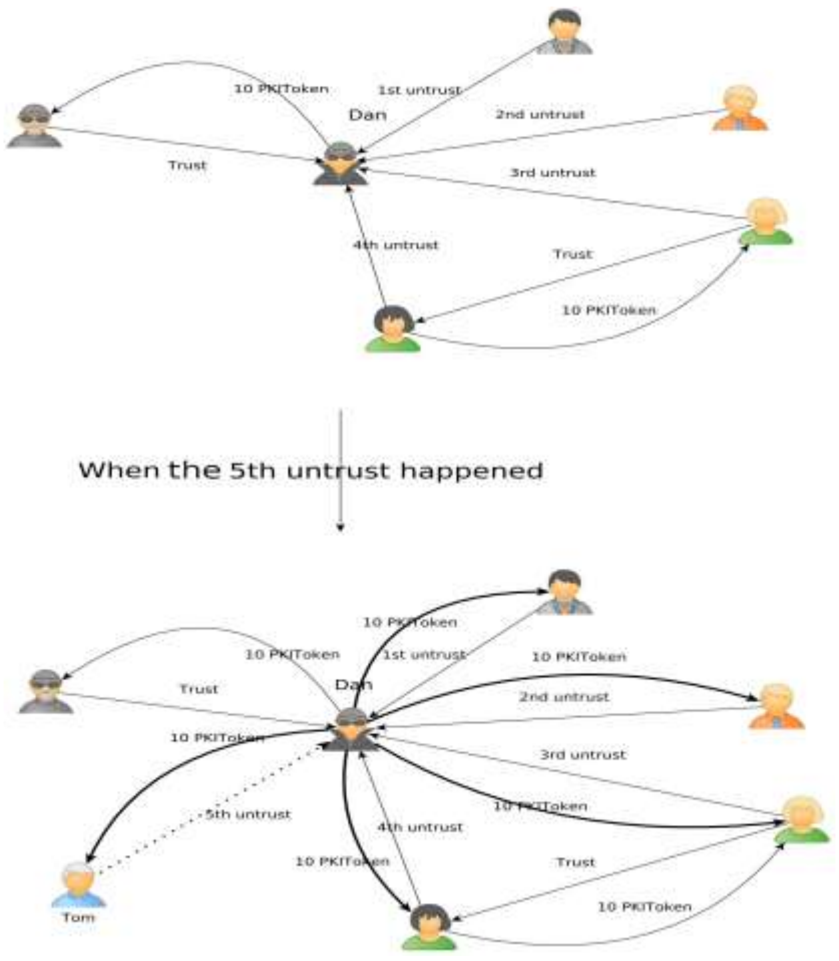


Fig. 1: ETHERST version 2.0: Untrust a node. When the 5th untrust happened, a punishment will be triggered. Allthe untrustors will receive an amount 10 PKIToken from the user who has been untrusted (Koa et al., 2021).

5. Development and deployment of dapp

In this section, we are going to explain the development stack and deployment steps for ETHERSTWEB.

To initiate the development of decentralised applications, the first step is to install a developmentframework. TruffleSuite (TruffleSuite, 2020), a globally recognised protocol suite, serves this purpose by functioning as both a testing framework and a communication medium with blockchain through the Ethereum Virtual Machine (EVM). TruffleSuite streamlines a developer's job by offering a toolset that compiles smart contracts into machine language, deploys them on public or private Ethereum networks, connects them to other contracts, and manages their binary code. With TruffleSuite's automated contract testing, developers can quickly build applications while locally simulating the Ethereum network on their computers. enabling the execution of transactions without incurring actual costs.

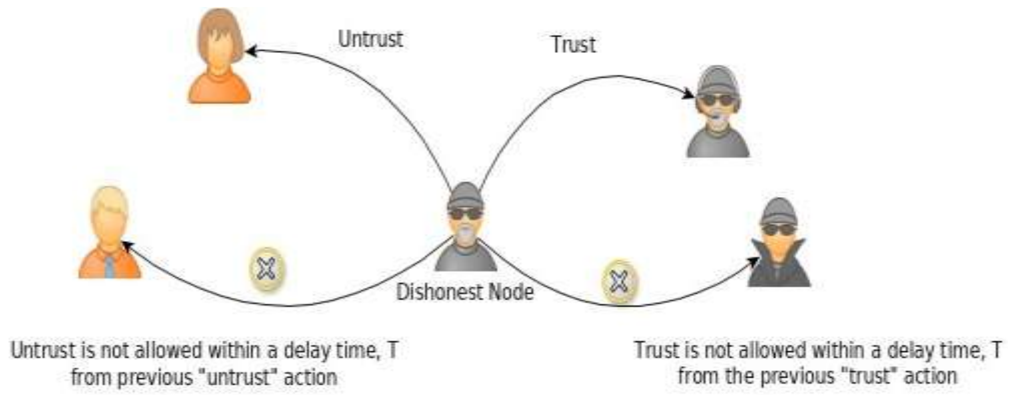

Untrust
Trust
Dishonest Node
Untrust is not allowed within a delay time, T from previous "untrust" action
Trust is not allowed within a delay time, T from the previous "trust" action

Fig. 2: ETHERST - Incremental delay between actions to prevent Sybil attack.

The second step in developing decentralised applications involves writing the smart contract using programming languages such as Solidity. Solidity is an object-oriented, high-level language used for developing smart contract source code that runs on the EVM, and it has syntax similar to JavaScript. It offers benefits of an object-oriented like inheritance and libraries that enable the creation of reusable code that can be called from various smart contracts. Writing a smart contract can be done using numerous tools, including a simple text editor, but one of the most commonly used is Visual Studio Code (VS Code). The developer writes the code for the smart contract using Solidity programming language and then compiles it into "EVM bytecode" so that the Ethereum Virtual Machine can understand it. Besides, web3.py or any web3 library is required for code development, as the code will act as an interfacebetween the frontend application and the Ethereum blockchain. The code's Application Binary Interface (ABI), which contains all the functions and features of the smart contract, is another critical element that enables interaction with the deployed smart contract in JSON format. This JSON format can be generated by the Solidity compiler once the smart contract code is built and compiled.

The third step to building the decentralised application is to develop the frontend application that interacts with the interface program developed in the second step. There are a few types of front-end applications that can be developed, web, mobile or desktop applications. For example, the decentralised application proposed in this paper, ETHERSTWEB, is a web application developed in Python programming language with Django framework (Foundation, 2005). Django web framework is a popular open-source web framework with the features of rapid development, security, fully loaded, scalable and versatile.

The fourth step in developing decentralised applications involves deploying the smart contract to a test network and conducting tests. Ganache which is known for its user-friendliness is used as the development test network. Ganache is bundled with TruffleSuite, essentially a personal blockchain that runs locally on a computer and is specifically designed for Ethereum development. By automatically configuring all the network settings and generating ten accounts, each holding 100 ETH, along with their corresponding mnemonics, Ganache enables the simulation of the vast Ethereum network. This feature allows developers to test smart contract operations and analyse account balances and transaction costs, facilitating experimentation with the balances of the ten accounts to simulate how the smart contract will function. As a result, developers can test the smart contract without incurring any costs and with much faster transaction execution speeds compared to those of the mainnet and public testnets such as Ropsten or Kovan.

The final step in developing a decentralised application is to deploy the entire application on the Ethereum mainnet, making it publicly available. However, before deployment, the application needs to be compiled and the developer must have the currency of Ethereum, Ethers in their account. The reason Ethers are needed is because deploying smart contract is a transaction in Ethereum and each transaction required a gas fee which needs to be paid in Ethers. Infura (Infura, 2016), a gateway and web3 provider, can be used to obtain Ethers and transfer the local executable to the main Ethereum network. Once deployed, the contract account address in the frontend part of the application must be updated, and the frontend must be deployed on a server or P2P network for fully decentralised. Monitoring the application after deployment can be done using blockchain explorers such as Etherscan, which is an analytical platform for public data on Ethereum. Etherscan allows real-time monitoring of transactions, blocks, wallet addresses, smart contracts, NFTs, and more. Figure 3, as seen below, by using ETHERSTWEB as an example, summarises all the steps required to develop and deploy an Ethereum decentralised application and shows some of the tools used for the development and deployment. The application was developed following the aforementioned methodology. The architecture of the application comprises three main Ethereum dapp layers:

- Application: This layer comprises the user interface and business logic that interacts with the smart contract. Communication with the Ethereum network is facilitated through an API, and the user interface may take the form of either a web application or a mobile application.

- Smart contract: The logic and rules of the dapp are defined in this layer, where smart contract code is written in Solidity programming language and deployed on the Ethereum network. Subsequently, the Ethereum Virtual Machine (EVM) executes the deployed smart contracts.

- Ethereum network: The layer comprises nodes that operate the Ethereum software and validate transactions, forming a distributed network responsible for maintaining the blockchain's integrity.

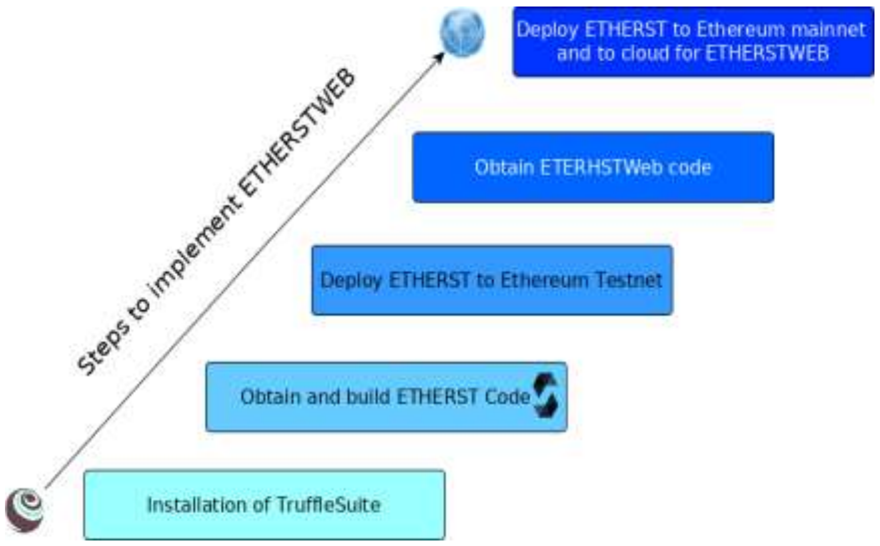

Steps to implement ETHERSTWEB
Deploy ETHERST to Ethereum mainnet and to cloud for ETHERSTWEB
Obtain ETERHSTWeb code
Deploy ETHERST to Ethereum Testnet
Obtain and build ETHERST Code
Installation of TruffleSuite

Fig. 3: Steps to implement ETHERSTWEB.

6. USING ETHERSTWEB FOR PROFILE VALIDATION BY COMMUNITY

To demonstrate the usage of ETHERSTWEB, we first landed at the home page of it as shown in Figure 4, we start with registering a user account. Figure 5 shows the sign-up page for ETHERSTWEB. Users need to input their username, email as well as password together with reconfirming password in order to successfully sign up. Whenever a user signs up successfully, ETHERSTWEB will create an account on the Ethereum blockchain which is also known as the wallet. At the same time, ETHERST will provide 50 PKIToken as an initial balance for the PKIToken which is needed for trusting and untrusting actions. To demonstrate the whole process, we start to register a user with the username "good".

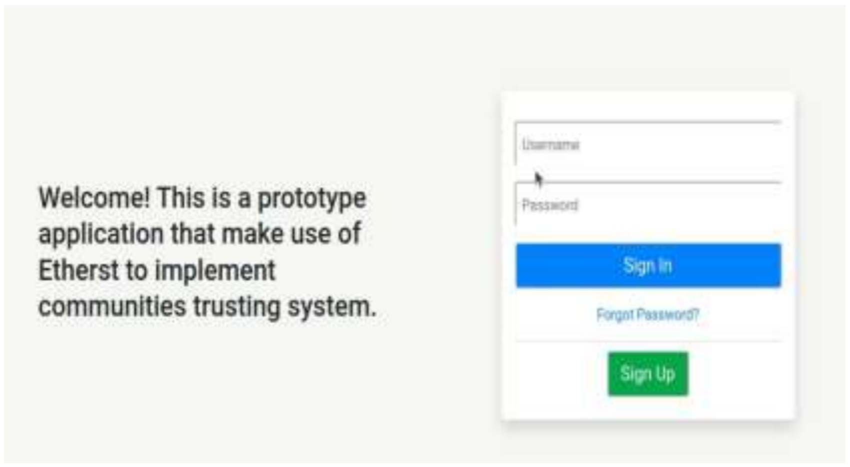

Welcome! This is a prototype application that make use of Etherst to implement communities trusting system.

Fig. 4: Home page of ETHERSTWEB.

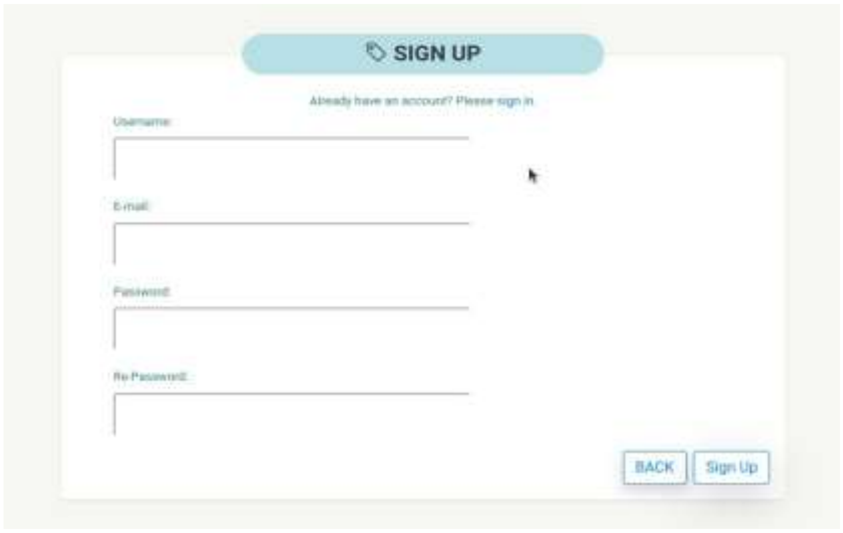
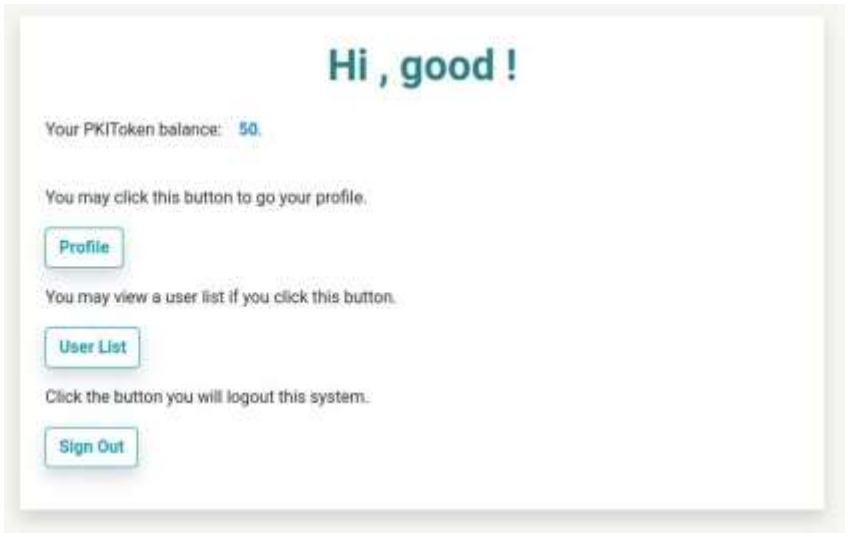


Fig. 5: Signup page of ETHERSTWEB.

After signing up, the user "good" will be redirected to the dashboard page where there are a few buttons to allow to link to other pages. Besides, the dashboard page will also show the PKIToken balance of the user as presented in Figure 6. All newly registered users will be allocated 50 PKIToken as the initial balance arranged by the ETHERST framework.


Hi , good !

Your PKIToken balance: 50.

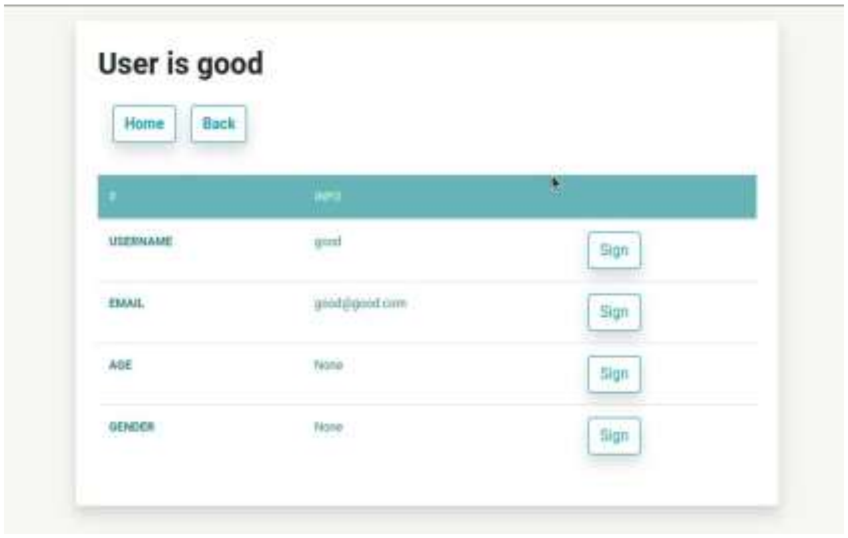
You may click this button to go your profile.

You may view a user list if you click this button.

Click the button you will logout this system.

Fig. 6: User's Dashboard page in ETHERSTWEB

From the dashboard page, user "good" is able to enter the profile page to update the profile data, once he confirms the data is correctly entered, he can "sign" on his own profile data by clicking the "sign" button as depicted in Figure 7 to certify the information provided is genuine. However, whether the information is real or fake is still unknown. Whenever a profile data is created, the frontend will make a call to the function "createAttribute" in ETHERST smart contract and create corresponding data inside blockchain. Meanwhile, the "sign" action in frontend will call the function "signAttribute" function in ETHERST smart contract.



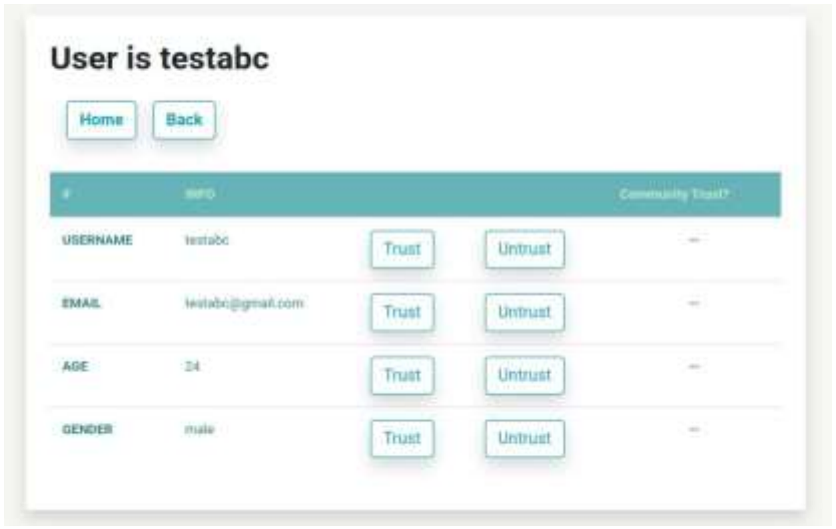
User is good

Home Back

#	INFO	
USERNAME	good	Sign
EMAIL	good@good.com	Sign
AGE	None	Sign
GENDER	None	Sign

Fig. 7: User's own profile page in ETHERSTWEB.

Once data is "signed", it is opened to the community to "trust" or "untrust" to certify the trustworthiness of this data. As shown in Figure 8, the user "good" opens the profile page of user "testabc", all the attributes are already signed, therefore, they are opened to the community to act to "trust" or "untrust". The community members can opt not to "trust" or "untrust" if they are unsure. In contrast, they can "trust" or "untrust" it if they really confirm data is real or is a fake information.



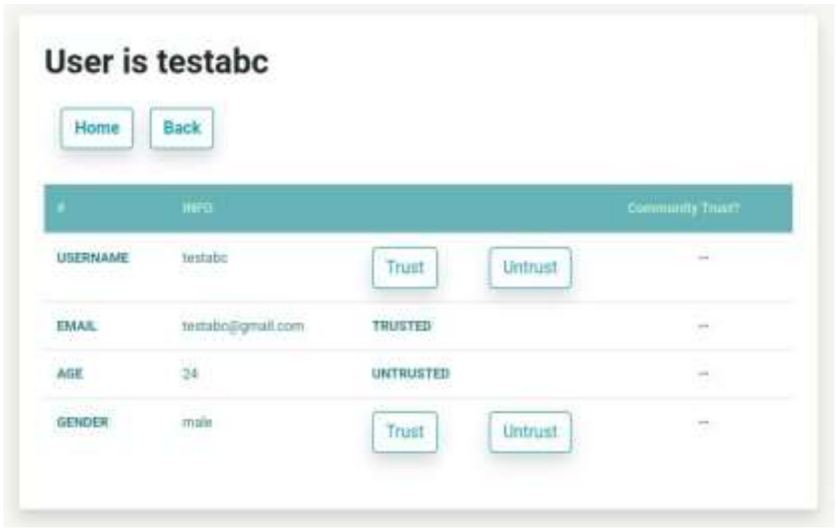
User is testabc

Home Back

#	INFO			Community Trust?
USERNAME	testabc	Trust	Untrust	–
EMAIL	testabc@gmail.com	Trust	Untrust	–
AGE	24	Trust	Untrust	–
GENDER	male	Trust	Untrust	–

Fig. 8: Other user's attributes which opened for trust or untrust action

If user "good" chooses to "trust" on the email attribute of user "testabc", ETHERSTWEB will make a call "trustSignature" on EHTHERST smart contract. ETHERST will run the internal algorithm and evaluate. A trusted attribute will be displayed as "TRUSTED" under the column INFO while an untrustedattribute will be shown as "UNTRUSTED" as depicted in Figure 9.



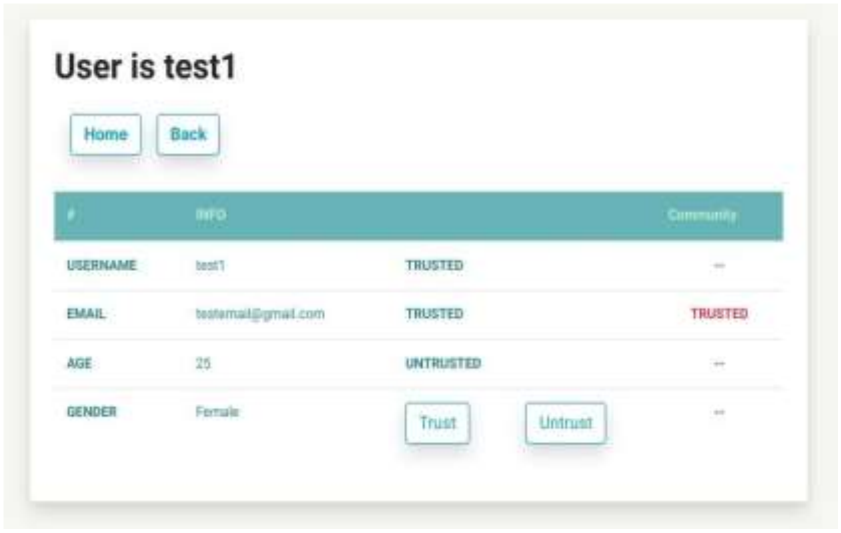
User is testabc

#	INFO			Community Trust?
USERNAME	testabc	Trust	Untrust	–
EMAIL	testabc@gmail.com	TRUSTED		–
AGE	24	UNTRUSTED		–
GENDER	male	Trust	Untrust	–

Fig. 9: Other user's attributes which are already trusted and untrusted

The column "Community Trust?" is a column that shows the result returned by ETHERST based on the internal algorithm whether this attribute is "trusted" or "untrusted" by the community members. A value of "–" means the ETHERST still does not have enough "trust" or "untrust" actions to compute the trustworthiness of this attribute.

Figure 10 shows the attributes of user "test1", and the attribute "email" is already "TRUSTED" by the community.



User is test1

#	INFO			Community
USERNAME	test1	TRUSTED		–
EMAIL	testemail@gmail.com	TRUSTED		TRUSTED
AGE	25	UNTRUSTED		–
GENDER	Female	Trust	Untrust	–

Fig. 10: An attribute already trusted by the community.

The community members can use the value in the column "Community Trust?" to evaluate the other users. In the case of an e-commerce application, buyers can review and evaluate the merchants with the community trusting results on their profile attributes, for example, "company name", or business behaviour attributes like "chat responsiveness" and "packing quality" which come from the review and rating subsystem. It will save the buyers from spending a lot of time to find out the trustworthiness of the information about the merchant to evaluate whether to deal with the merchant.

CONCLUSIONS AND POTENTIAL FUTURE WORK

In a nutshell, false information floods the Internet nowadays and some false information can create chaos in our harmonious society. The authority is spending a lot of resources validating the information and censoring them to keep people from fake news and wrong information. The Internet is built by the community and for the community. The proposed ETHERSTWEB, let the community members help in combating false news and inaccurate information. By incorporating the ETHERSTWEB features, we encourage the community members to do self-protection and guard the accuracy of the information. Indirectly, it will increase people's awareness of the impact of fake news and information on society.

ETHERSTWEB is currently developed as a Django project instead of a Django app. One of the improvements that can be done is to develop it as a reusable Django app, that is pluggable to any other Django projects to have the built-in information trustworthiness management by community members. It can also be developed as a microservice component in a microservice architecture (Liu et al., 2020) and itcan be used to extend any existing application functionality regardless of the programming language or framework used. Finally, the source code of ETHERSTWEB can be found on GitHub, and it is released under the MIT license and can be accessed at https://github.com/cyberkoa-mmu/etherstweb.

ACKNOWLEDGEMENTS

This work was supported by the Multimedia University's GRA scheme and the Telekom Malaysia Research & Development Grant (RDTC/221045).

REFERENCES

[1] Aldwairi, M., & Alwahedi, A. (2018). Detecting fake news in social media networks. Procedia Computer Science, 141, 215-222. Retrieved from https://www.sciencedirect.com/science/article/pii/S1877050918318210 (The 9th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN-2018) / The 8th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH-2018)/ Affiliated Workshops) doi: https://doi.org/10.1016/j.procs.2018.10.171

[2] Bahri, L., & Girdzijauskas, S. (2018). When trust saves energy: A reference framework for proof of trust (pot) blockchains. In Companion proceedings of the the web conference 2018 (p. 1165–1169). Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee. Retrieved from https://doi.org/10.1145/3184558.3191553 doi: 10.1145/3184558.3191553

[3] Buterin, V. (2013). Ethereum: A next-generation smart contract and decentralized application platform.

[4] De', R., Pandey, N., & Pal, A. (2020). Impact of digital surge during covid-19 pandemic: A viewpoint on research and practice. International Journal of Information Management, 55, 102171. Retrieved from https://www.sciencedirect.com/science/article/pii/S0268401220309622 (Impact of COVID-19 Pandemic on Information Management Research and Practice: Editorial Perspectives) doi: https://doi.org/10.1016/j.ijinfomgt.2020.102171

[5] Dhall, S., Dwivedi, A. D., Pal, S. K., & Srivastava, G. (2021, nov). Blockchain-based framework for reducing fake or vicious news spread on social media/messaging platforms. ACM Trans. Asian Low- Resour. Lang. Inf. Process., 21(1). Retrieved from https://doi.org/10.1145/3467019 doi: 10.1145/3467019

[6] Douceur, J. R. (2002). The sybil attack. In P. Druschel, F. Kaashoek, & A. Rowstron (Eds.), Peer-to peer systems (pp. 251–260). Berlin, Heidelberg: Springer Berlin Heidelberg.

[7] ERC-20. (2021, June). Eip-20: Erc-20 token standard. Retrieved from https://eips.ethereum.org/EIPS/eip- 20

[8] Jam, F. A., Singh, S. K. G., Ng, B., & Aziz, N. (2016). Effects of Uncertainty Avoidance on Leadership Styles in Malaysian Culture, , International Journal of Advance Business and Economics Research, 14(8), 7029-7045.

[9] Foundation, D. S. (2005). Django. Computer software. Retrieved from https://www.djangoproject.com/ Infura. (2016). Infura. Online. Retrieved from https://infura.io/

[10] Jaroucheh, Z., Alissa, M., Buchanan, W. J., & Liu, X. (2020). Trustd: Combat fake content using blockchain and collective signature technologies. In 2020 ieee 44th annual computers, software, and applications conference (compsac) (p. 1235-1240). doi: 10.1109/COMPSAC48688.2020.00-87

[11] Koa, C.-G., Heng, S.-H., & Chin, J.-J. (2021). Etherst: Ethereum-based public key infrastructure identity management with a reward-and-punishment mechanism. Symmetry, 13(9). Retrieved from https://www.mdpi.com/2073-8994/13/9/1640 doi: 10.3390/sym13091640

[12] Liu, G., Huang, B., Liang, Z., Qin, M., Zhou, H., & Li, Z. (2020). Microservices: architecture, container, and challenges. In 2020 ieee 20th international conference on software quality, reliability and security companion (qrs-c) (p. 629-635). doi: 10.1109/QRS-C51114.2020.00107

[13] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system.

[14] Qayyum, A., Qadir, J., Janjua, M., & Sher, F. (2019, jul). Using blockchain to rein in the new post-truth world and check the spread of fake

news. IT Professional, 21(04), 16-24. doi:10.1109/MITP.2019.2910503

[15] Huiza, D. A. M.-., Beru, A. C., Apaza, S. L. V., Choque, M. E. C.-., & Yucra, W. P. (2023). Impacts of Supervision, Information and Communication between Management Control and Administrative Management Procedures. International Journal of Membrane Science and Technology, 10(3), 563-575. https://doi.org/10.15379/ijmst.v10i3.1573

[16] Ryabitsev, K. (2014, February). Pgp web of trust: Core concepts behind trusted communication.

[17] Sengupta, E., Nagpal, R., Mehrotra, D., & Srivastava, G. (2021). Problock: a novel approach for fake news detection. Cluster Computing, 24(4), 3779–3795. Retrieved from https://doi.org/10.1007/s10586- 021-03361-w

[18] TruffleSuite. (Oct. 2020). Trufflesuite: Sweet tools for smart contracts. Retrieved from https://www.trufflesuite.com

[19] Vicario, M. D., Bessi, A., Zollo, F., Petroni, F., Scala, A., Caldarelli, G., Quattrociocchi, W. (2016). The spreading of misinformation online. Proceedings of the National Academy of Sciences, 113(3), 554-559. Retrieved from https://www.pnas.org/doi/abs/10.1073/pnas.1517441113 doi: 10.1073/pnas.1517441113

[20] Wahane, A., & Patil, B. (2022). Blockchains to curb fake news in an online world. In 2022 international conference for advancement in technology (iconat) (p. 1-6). doi: 10.1109/ICONAT53423.2022.9725933

DOI: https://doi.org/10.15379/ijmst.v10i1.1819