UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL DE CALCULATOARE

PROIECT DE DIPLOMĂ

Evenimente in Trafic Bazate pe Blockchain

Raoul-Renatto Sulimovici

Coordonator științific:

Prof. Dr. Ing. Ciprian Dobre

BUCUREȘTI
2022

UNIVERSITY POLITEHNICA OF BUCHAREST
FACULTY OF AUTOMATIC CONTROL AND COMPUTERS
COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

DIPLOMA PROJECT

Blockchain-Based Traffic Events

Raoul-Renatto Sulimovici

Thesis advisor:

Prof. Dr. Ing. Ciprian Dobre

BUCHAREST
2022

CONTENTS

SINOPSIS

Confidențialitatea utilizatorilor este un subiect cu o deosebită importanță în folosirea Internetului. Există multe scenarii în care acest lucru este esențial, însă lucrarea pune accent pe unul specific: afișarea locației unei persoane către o aplicație. Mai exact, afișarea locației unui utilizator către aplicațiile de navigare în trafic (Google Maps, Waze). Scopul acestui studiu este să demonstreze utilitatea anonimității utilizatorilor combinată cu îmbunătățirea acurateței evenimentelor din trafic, în cazul aplicațiilor de acest tip. Confidențialitatea va fi obținută pe baza unei rețele de tip blockchain, care va reține datele evenimentelor din trafic, acestea fiind direct influențate de către utilizatori. Blockchain-ul reprezintă un registru distribuit de tranzacții imutabile efectuate într-o rețea. Tranzacțiile pot fi văzute de către orice participant implicat în validarea acestei rețele și sunt asociate cu o adresă în rețea (un șir de caractere), decuplandu-se astfel identitatea reală a unui utilizator de o tranzacție. Rezultatele studiului conduc către o îmbunătățire majoră a sistemelor de navigație, prin folosirea tehnologiei blockchain, alături de un sistem bazat pe reputație pentru a oferi credibilitate și siguranță participanților la trafic.

Cuvinte cheie: Blockchain, Sistem bazat pe reputație

ABSTRACT

Users' Privacy is a subject with a particular importance when browsing the Internet. There are multiple scenarios in which this is essential, but the thesis will emphasize one in particular: disclosure of a person's location to an application. More specifically, disclosure of a person's location to a traffic navigation application (Google Maps, Waze). The purpose of this study is to demonstrate the utility of user anonymity combined with an improvement in traffic events accuracy, in this type of applications. Privacy will be ensured based on a blockchain type network which will retain the data of the traffic events, that are directly influenced by the users. Blockchain is a distributed immutable ledger of transactions being made in a network. The transactions can be seen by any participant involved in the validation of this network and are associated with an address in the network (a string of characters), thus decoupling the real identity of a user from a transaction. The results of the study lead to a major improvement of navigational systems, by using blockchain alongside a reputation-based system in order to ensure credibility and safety for all the traffic participants.

Keywords: Blockchain, Reputation-based system

1 INTRODUCTION

This paper presents an improvement to accustomed navigation applications, by guaranteeing users' privacy and increasing their trust into the traffic events [1, 2, 3].

The motivation for this project is the fact that data is becoming the most valuable asset in the world and with its rapid increase in size, people's interest in keeping their data private and secure is also increasing [1]. Humans tend to spend substantially more time on the Internet as each year passes and so, finding a method of assuring them in regards to what happens to their information and who sees it becomes essential.

In addition to giving users a feeling of safety regarding their data, this proposal aims to reduce the effect of wrongdoers, by taking into consideration their past behavior when they are trying to influence the traffic [3].

In order to achieve the goals mentioned, the solution incorporates a blockchain network, which allows us to store data in a permanent, transparent and tamper-proof way, while keeping users' identity anonymous through blockchain random addresses, alongside a reputation-based system, which lets users introduce or confirm the presence of events in traffic, only if they are entitled to, with respect to the reputation they own [1, 2, 4, 5, 3].

Results show that the objectives have been accomplished and, in addition to that, the final product reveals great performances in terms of memory consumption, security and gas fees.

The thesis is structured in 6 chapters: Introduction, Related Work, Proposed Solution, Implementation Details, Results and Evaluation and Conclusion. Related Work illustrates the differences and similarities between this approach and other already formulated ideas. Proposed Solution gives more details about the technologies and how they were embedded from an architectural point of view. Implementation Details presents more in-depth information about the logic involved. Results and Evaluation highlights the results and experiments conducted, but also the strengths and weaknesses of the outcome. The Conclusion sums up everything that has been mentioned whilst guiding those interested to what could have been done better.

2 RELATED WORK

Some proposals ensuring privacy and credibility while using traffic applications have already been made. In this chapter, the current state-of-the-art will be presented and compared with the solution of this paper.

In Reference [1], the solution proposed includes a custom and personal blockchain network that keeps information about locations, speed and traffic events. The privacy is guaranteed by a set of security policies, managed by each user for its own purpose. The policies control speed sharing, location sharing and additional data sharing based on the previous two. An improvement could be excluding the name of a user from its database table definition. This name could lead to a real person, thus violating privacy. The consensus is accomplished based upon users' input, but there is no method of taking into account a contributor's credibility. Assigning a credibility value (reputation system) to an answer would result in better accuracy for the events.

Reference [2] presents a solution similar to the one in this paper. It consists of a blockchain network and a reputation-based system. The blockchain is divided into three types of nodes: User Node (which represents a user that reports its verdict regarding events), Region Node (which divides answers in regions and takes into account the User Node's reputation) and Master Node (secure node that gathers all information). The implementation allows a user to get directions of travel, find out speed alerts and to share speed and traffic events. The reputation for a specific event is computed by subtracting the number of wrong answers from the number of correct answers and dividing by the number of total answers per user, in the last minute. The paper also indicates the memory impact and methods of better memory management for a similarly designed application.

The improvement added by the reference [4] is that of a new type of consensus protocol. It is called "Proof of Pseudonym" and it embodies some parts of the earlier consensus protocols, alongside with a Pseudonym Shuffling Algorithm and a faster execution time. Basically, a random set of pseudonyms is cyclically assigned to each user after an expiration date. Consensus is achieved based on a mining process which is won by a random validator. Another difference is the usage of "Road Side Units", which represent fixed infrastructure endpoints that communicate with the vehicles, rather than a smartphone, in the hands of a driver, communicating with a central endpoint, as presented in this paper. This makes this solution stand out in terms of energy consumption and even time complexity, the drawback being a diminished accuracy for the presence of traffic events.

Reference [5] has a different objective than the other articles mentioned, although it still emphasizes the use of blockchain technology in Intelligent Transport Systems. This solution saves information about: vehicles (speed, location), weather conditions, congestion and accidents. All this data is then used to serve multiple purposes, the main two being: as evidence when stating the conditions in which an accident has occurred and as a mean of predicting the incoming road congestions and delays.

Reference [3] took inspiration from reference [1] and improved the proposal by adding reliability on the events with a reputation system. Differences between this paper's solution and the one in reference [3] were found in terms of system design and implementation details, while the applied technologies are fundamentally very similar.

Table 1 shows a comparative analysis of related work and the inspiration for it comes from reference [3].

Table 1: Comparative analysis of related work

Reference Work	Advantages and disadvantages		
	Guaranteed Confidentiality	Use reputation metric in traffic event validation	Easy implementation in real-life scenario
[1]	yes	no	yes
[2]	yes	yes	no
[4]	yes	no	yes
[5]	no	no	yes
[3]	yes	yes	yes
Blockchain-Based Traffic Events	yes	yes	yes

3 PROPOSED SOLUTION

This chapter acts as an introduction to the technologies used and to the solution developed from an architectural point of view.

3.1 Theoretical Basis

3.1.1 Blockchain

Blockchain technology represents a network of nodes, connected to each other, with each node generating data and helping reach consensus [2]. More clearly, blockchain acts as a digital, immutable, distributed and usually public ledger that stores data into blocks [1, 4]. The entirety of blocks holds records of all ever-executed transactions, saved permanently into the network in a tamper-resistant way [1, 2].

The term "blockchain" comes from the inside structure. Every block is linked to its parent, in chronological order, like a "chain of blocks", by a field that contains the hash of the previous block [1]. Each node, whether it is trustworthy or malicious, holds a copy of the whole blockchain, which is kept up to date and secure based on cryptographic algorithms and the computing power generated by the network [1, 2].

The key factors that make blockchain such an interesting technology are the ability to identify nodes only by their hexadecimal address, therefore keeping the identity anonymous and the fact that tampering the data is nearly impossible, due to the fact that you need to change all the blocks that come after the one tampered and that computational power is really hard to obtain [1].

Figure 1 is taken from reference [6] and illustrates an example of a blockchain.

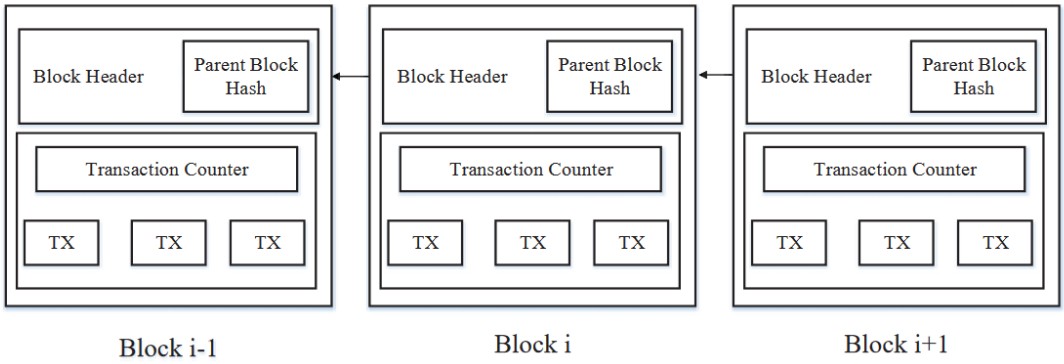

Figure 1: Blockchain Example

3.1.2 Reputation System

A reputation system is one in which past behavior is taken into account when evaluating a user's credibility. A user's permissions should be changed accordingly to its reputation within the system or within the community of users [2, 3].

3.2 System Design

The main purpose of the application is to show the traffic events that are happening in traffic whilst preserving user anonymity and maximizing events presence accuracy [2, 4]. Before adding an event to the blockchain, it first has to be introduced to the database as a request and, during a period of time, to be subjective to confirmation or rejection from the other users nearby. After an evaluation period of time, the request can become an event and can be introduced further into the blockchain for permanent storage.

The solution involves:

- a Blockchain module
- a Digital wallet available as a browser extension that sends transactions to the blockchain
- a Database module useful for retaining event requests and the reputation system
- a Client module in order for the users to interact with the Server Module and the wallet
- a Server module that facilitates communication between Client and Database

Figure 2 shows the System Design diagram.

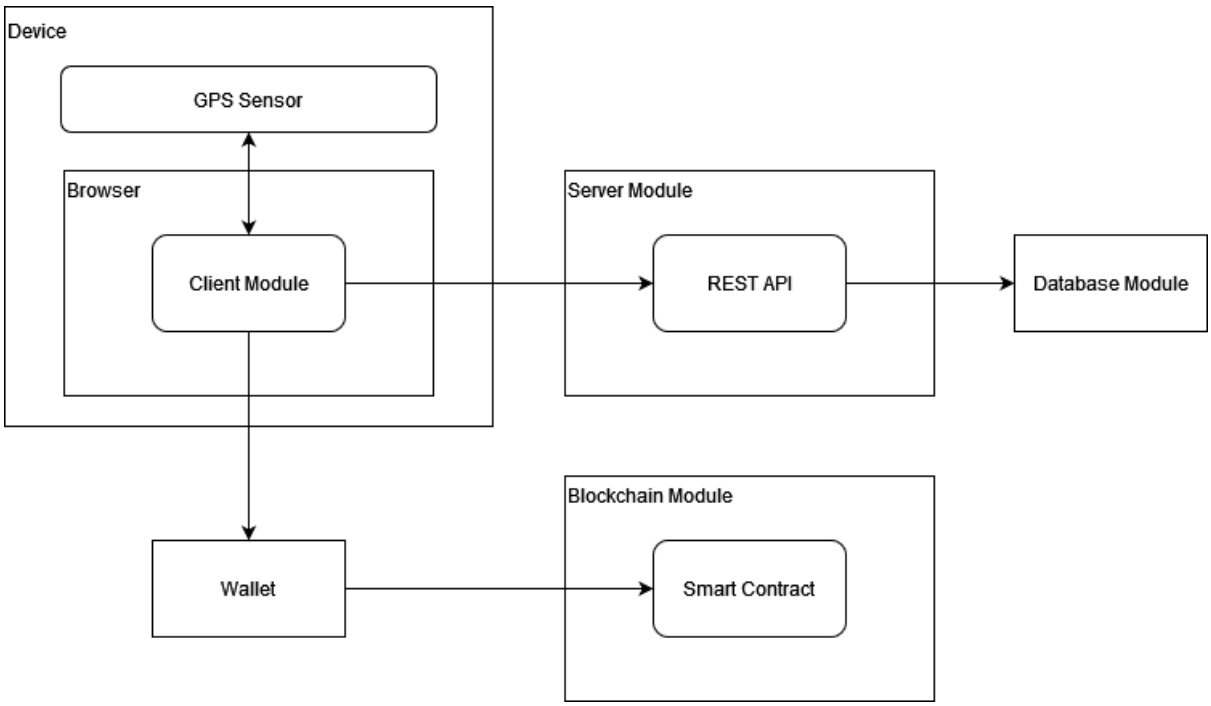

Figure 2: System Design

3.2.1 Blockchain Module

This module is built on Ethereum[1] network and implements a smart contract. The contract saves a list of events, each containing the following fields:

- User Address
- Event Type
- Location
- Date
- Speed

The methods associated with the smart contract are:

- Create Event
- Get Event
- Delete Event

3.2.2 Digital Wallet

The wallet will receive the desired transaction and, based upon the current balance and destination address, it will either fulfill the transaction and return a success message or reject it and return an error message.

[1] `https://ethereum.org/en/`

3.2.3 Database Module

The database stores information about users and the received requests.

Figure 3 presents the Database Schema.

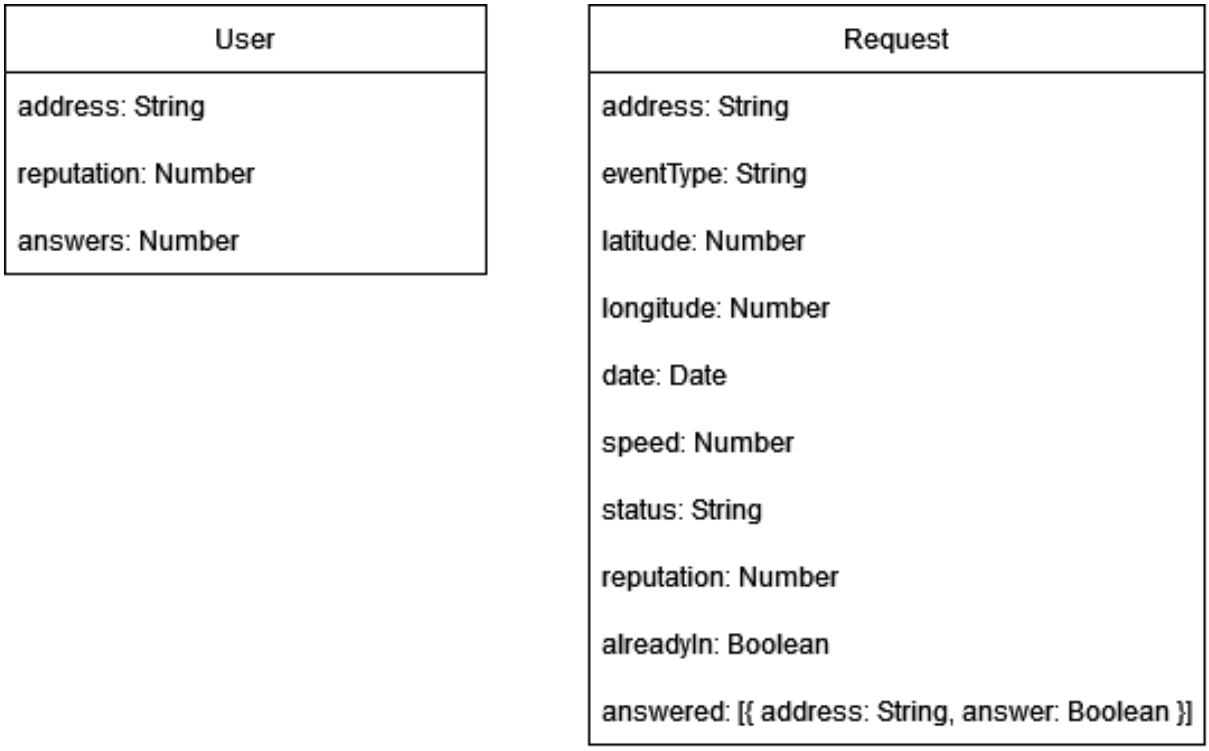

Figure 3: Database Schema

3.2.4 Server Module

The Server Module contains a REST API that responds to HTTP requests from the Client Module and queries the database. The reputation logic is also included in the Server Module. Each user has a reputation assigned, which will then be passed to each request he makes. The actions that triggers a change in a user's reputation will be detailed in the next chapter.

3.2.5 Client Module

The Client exposes a Web Application that lets the user do the following actions: see the current requests and the active traffic events, add a new request to the database and confirm or reject requests if the user's location is close to the event's area.

4 IMPLEMENTATION DETAILS

This chapter will cover the overall logic of the solution and, according to the System Design (Figure 2) and the previously mentioned information, will thoroughly explain each of the 5 modules present.

4.1 Blockchain Module

The blockchain is developed using Solidity[1] as a programming language and Truffle Suite[2] as a development environment. Ganache[3], a software tool offered by Truffle, is used to run the personal blockchain network locally. Everything will be based on Ethereum Virtual Machine[4].

The smart contract deployed on the blockchain handles data storing and retrieval for the traffic events.

The contract exposes an integer, signify the current number of events. The events mapping is private and accessible through the methods. As stated in the previous chapter, a saved event contains the fields: eventType, date, latitude, longitude, speed, owner.

The available methods are:

- getEvent, which returns an event at a specific index
- createEvent, which adds a new event to the mapping and, implicitly, to the blockchain
- deleteEvent, which invalidates the entry in the blockchain by deleting it from the mapping

All these functionalities are usable from the Client Module, through the Digital Wallet.

[1] https://docs.soliditylang.org/en/v0.8.15/
[2] https://trufflesuite.com/
[3] https://trufflesuite.com/ganache/
[4] https://ethereum.org/en/developers/docs/evm/

4.2 Digital Wallet

The wallet used in the project is MetaMask[5]. It is available in the form of a browser extension or as a mobile app. In order to use it, you have to create or to import an account into the wallet. The account will contain your balance and the wallet allows you to buy, send and swap different currencies. Before being added to the blockchain, every transaction must pass through the wallet and balance must be checked. Then, a user can confirm or reject the transaction, acknowledging the gas fees.

Figure 4 illustrates MetaMask appearance.

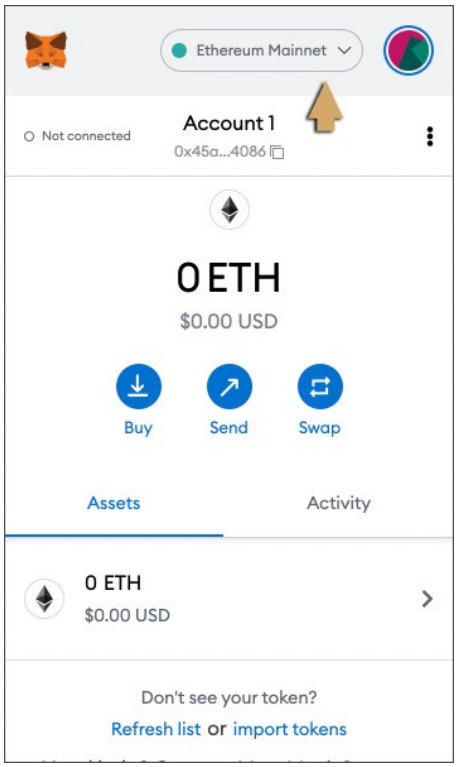

Figure 4: Metamask[6]

4.3 Database Module

The Database Module has been implemented in MongoDB[7] with the aid of an ORM (Object-Relational Mapping)[8] called "mongoose". The ORM is specific to MongoDB and Javascript. The models defined respect the schema that can be seen in Figure 3

[5] https://metamask.io/
[6] https://medium.com/klaytn/how-to-add-klaytn-to-metamask-b3bdd970c0e8
[7] https://www.mongodb.com/
[8] https://www.techopedia.com/definition/24200/object-relational-mapping--orm

4.4 Server Module

The server is implemented in Javascript[9], NodeJS[10] and ExpressJS[11].

The routes available are as follows:

- Get Requests
  Retrieve all the requests from the database

- Add a new request
  In order for the request to be added to the database, it has to be at minimum 50 meters away from any other request or event and to be at minimum 500 meters away from any other request or event that has the same type. If the user does not have an entry in the User database table, it will be added.

- Add an answer to a request
  Adds a confirmation or rejection to a request in the database.

- Reset Request
  Change an existing and confirmed request's status back to Pending in order to reevaluate its presence.

- Update Request Status
  Update the request's status accordingly and change involved users' reputation

4.5 Client Module

The Client is developed using React[12] v17.

The workflow of the application starts with the connection to the client app using a web browser. In order to access it, one needs to turn on the wallet and assign an account for usage. After that, as seen in Figures 5 and 6, a user finds a map centered on its current location that is showing the requests (which can be answered) and the active events alongside with an additional button, in order for the user to submit the his own requests. Each event or request has a corresponding color: Traffic Jam - Orange, Police - Blue, Accident - Red, Road in construction - Green.

[9] https://developer.mozilla.org/en-US/docs/Web/JavaScript
[10] https://nodejs.org/en/
[11] https://expressjs.com/
[12] https://reactjs.org/

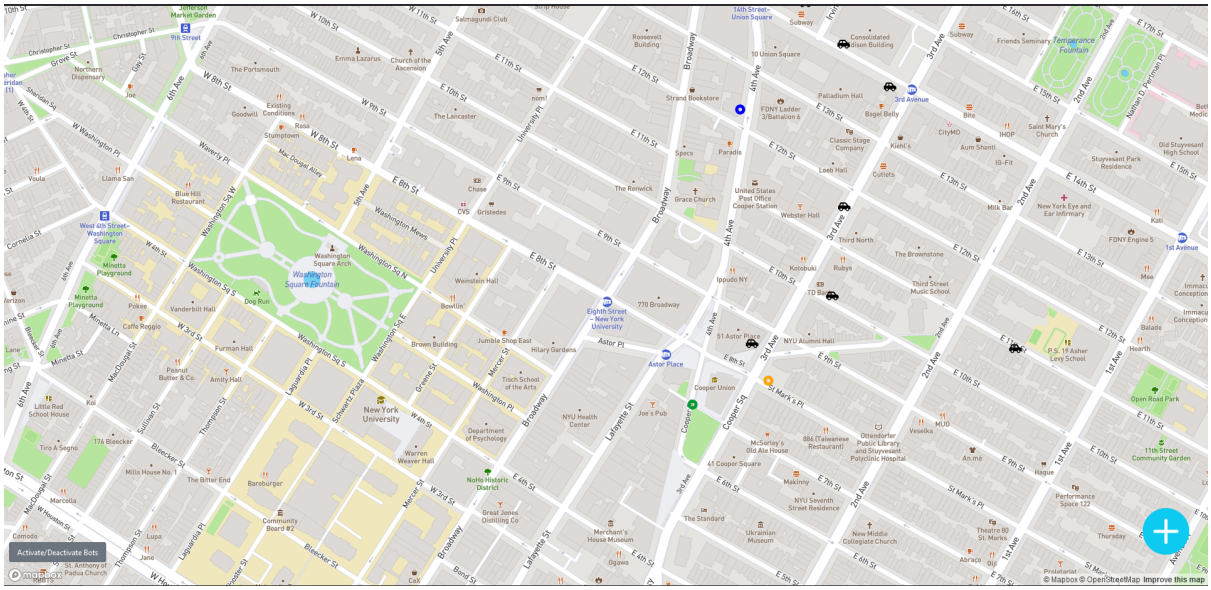

Figure 5: Application Screenshot

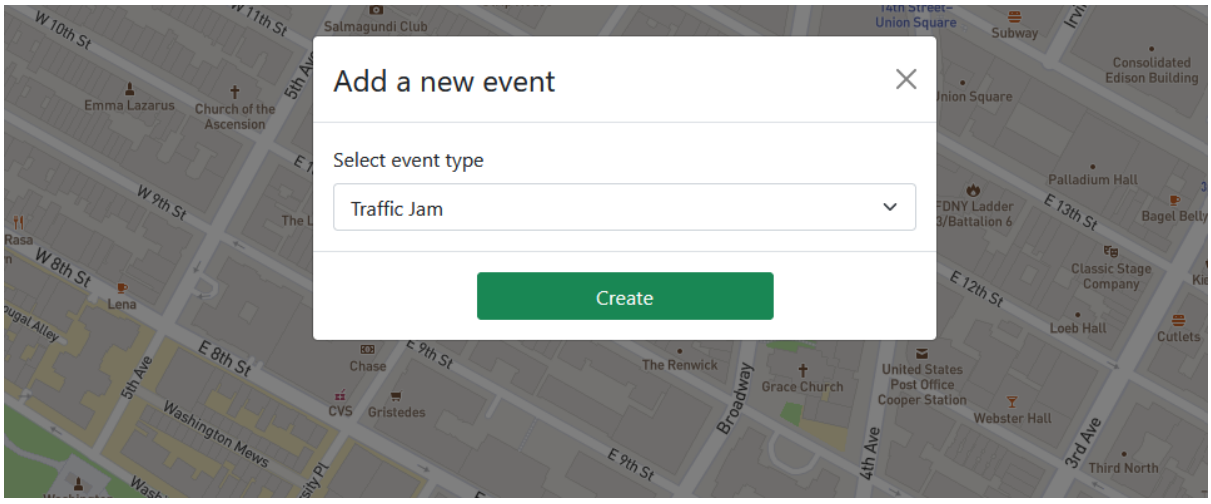

Figure 6: Request Addition Modal

Figure 7 illustrates the information card alongside with the confirmation and rejection buttons.

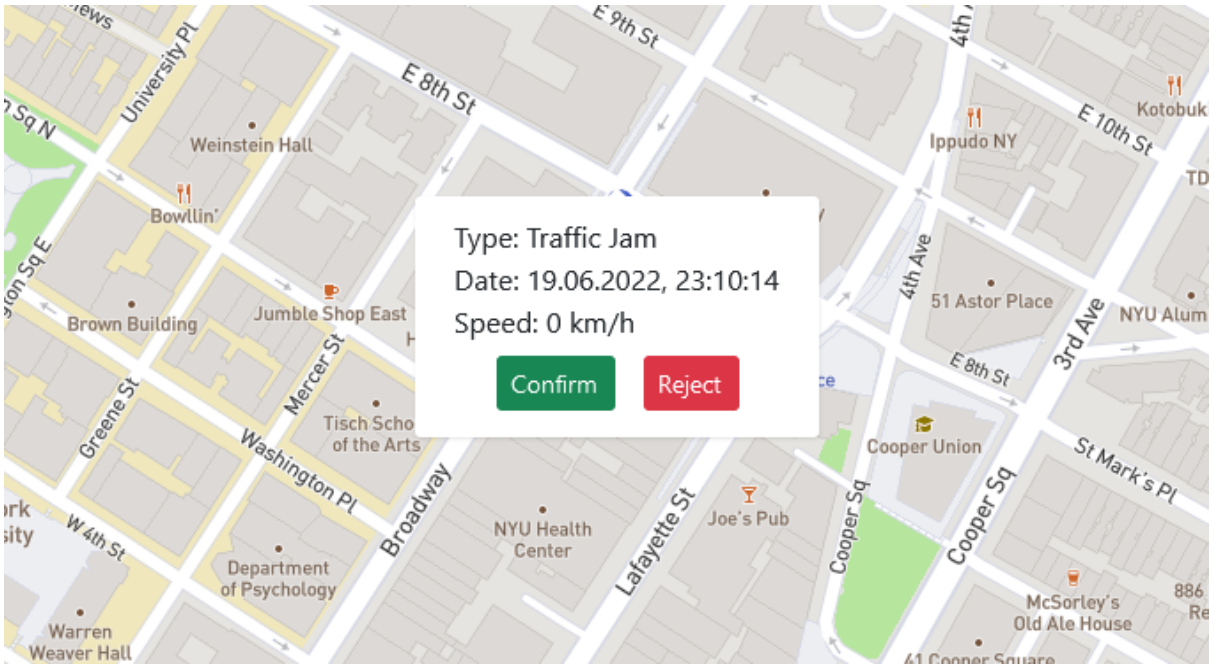

Figure 7: Request Answering Card

4.5.1 Request Adding

The "Create" button inside the Request Addition Modal (Figure 6) will be enabled only if a request or event is not nearby. In order for the button to work, no event or request has to be in less than 50 meters and no event or request of the same type has to be in less than 500 meters.

4.5.2 Bots

There also is a "Activate/Deactivate Bots" button in the bottom-left corner, which activates a set of fictive users that demonstrate how a user would see the other traffic participants. The bots are randomly driving around the user's location, and, once in 15 seconds, a random bot is trying to add a new request and all of them are answering nearby requests, with a 50% chance of a confirmation or a rejection.

4.6 Additional Information

4.6.1 Request Evaluation

When a user answers a request, the evaluating buttons will disappear, thus letting every user confirm or reject a request only once per evaluation round. The evaluation is being made by the logic in Algorithm 1.

Algorithm 1 Request Evaluation Algorithm

```
1:  procedure EVALUATE(answersArray, ownerReputation)
2:      totalWeight ← sum(answersArray.reputation)
3:      acceptWeight ← sum(answersArray.reputation), where answer value is true
4:      refuseWeight ← totalWeight - acceptWeight
5:      refusedThreshold ← 2
6:
7:      if acceptWeight ≥ totalWeight / 2 + 1 then
8:          addToBlockchain()
9:          updateDatabase()
10:         increaseOwnerReputation()
11:         updateValidatorsReputation()
12:     else if acceptWeight ≥ refuseWeight - refusedThreshold then
13:         addToBlockchain()
14:         updateDatabase()
15:         updateValidatorsReputation()
16:     else
17:         updateDatabase()
18:         decreaseOwnerReputation()
19:         updateValidatorsReputation()
```

4.6.2 Reputation Update

Subsequently to the evaluation, the database will be updated accordingly and, in case of acceptance, a transaction will be sent to the digital wallet which will then communicate with the smart contract and the Blockchain Module.

The initial reputation is 5 for every registered user. The maximum reputation a user can achieve is 10 and the minimum is 0. In the case of a request's evaluation, the involved users' reputation will be updated as follows:

- If the request is confirmed, the request owner's reputation will increase, otherwise, if the request is rejected, it will decrease
- For all the users that answered a request, if their answer differs from the request's outcome, their reputation will decrease

The formula for reputation update is:

$$answersWeight = max(userAnswers,\ 1) * sign\ /\ totalAnswers \tag{1}$$

$$newReputation = min(max(oldReputation + answersWeight, 0), 10) \tag{2}$$

The equation above will be applied to each involved user.

More explanations for the formula can be seen in Algorithm 2.

Algorithm 2 Reputation Update Algorithm

```
1: procedure updateReputation(user, request)
2:     oldReputation ← user.reputation // save current reputation
3:     userAnswers ← max(user.answers, 1) // save number of answers for the user
4:     totalAnswers ← getTotalAnswers() // total number of answers, given by all users
5:
6:     sign ← 0
7:
8:     // If it is the owner of the request
9:     // Increase reputation if request is confirmed, decrease otherwise
10:    // Decrease it otherwise
11:
12:    if isOwner(request, user) then
13:        sign ← isConfirmed(request) ? 1 : -1
14:    else // User that answered
15:        sign ← answerSameAsOutcome(request, user) ? 0 : -1
16:
17:    answersWeight ← userAnswers * sign / totalAnswers
18:
19:    newReputation ← oldReputation + answersWeight
20:    newReputation ← max(newReputation, 0)
21:    newReputation ← min(newReputation, 10)
22:    // Limit result to [0,10] interval
23:
24:    return newReputation
```

4.6.3 Events Refresh and Expiration

The requests and events will be queried every 15 seconds from the database and the blockchain. The evaluation time for each request is 5 minutes. After those 5 minutes the outcome of the request, based on the answers, will be updated into the database and, if applicable, the event added to the blockchain.

Every event introduced in the blockchain has an expiration date, based on its type. Their expiration rates are:

- 20 minutes for a Police event
- 30 minutes for an Accident event
- 1 hour for a Traffic Jam event
- 2 hours for a Road in Construction event

After an event has been removed from the blockchain, according to its expiration date, it will be reintroduced in the database as a pending request. If that is the case, the owner's reputation will not be updated again, as this event has already been introduced and evaluated, but further wrong answers will still decrease the reputation for the involved users. In order to accomplish this, the boolean field in the Requests Schema called "alreadyIn" will be set to true. (Figure 3)

5 RESULTS AND EVALUATION

In order to evaluate the solution results, an experiment has been conducted. The experiment consists of a simulated usage of the application, with 8 bots activated, that are mimicking the behavior of real users. One of them is publishing a request every 15 seconds and they are all answering their nearby events. The probability of answering an event with a confirmation or rejection is 50%. The number of requests they were allowed to introduce in the database for this experiment was limited to 50.

During the experiment, the actual number of events that were, at least once, introduced in the blockchain is 30, which equals 60% of requests.

Figures 8 and 9 show the dependency between user answers and reputation for each of the involved users.

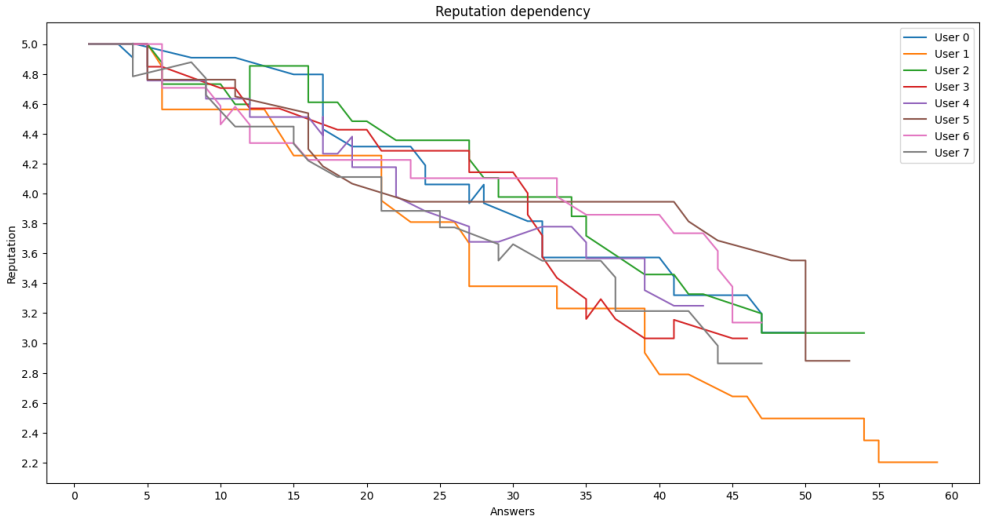

Figure 8: Reputation-Answers Dependency Combined

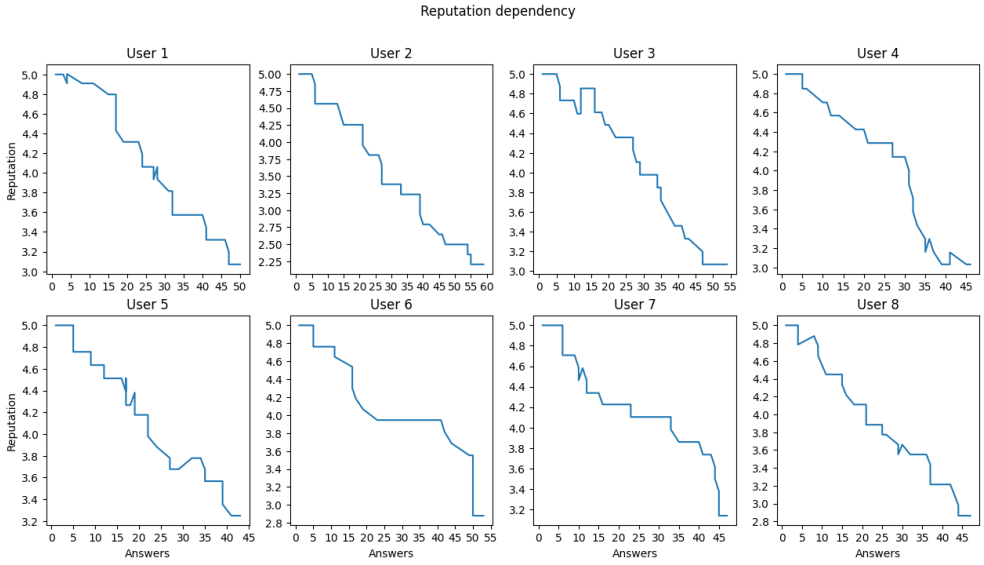

Figure 9: Repuation-Answers Dependency Individualized

5.1 Possible Threats

Even though the proposed solution concentrated on maximizing the security, there are still some attacks and threats that are present, which will be detailed in the next sections.

5.1.1 Malicious Users

What this threat means is that a group of users might connect to the application and purposely add content that is not valid. In order to bypass this issue, we proposed the Reputation System. This feature ensures that without requests confirmation, a user can not tamper the data involved. If they were to introduce fake data or answer incorrectly to the events present, they would very shortly get a decrease in reputation, therefore making them untrustworthy and their answers would become less significant.

Figures 8 and 9 demonstrate that in the case of dishonest behavior, the attackers reputation will tend to decrease to a minimum.

The risk that this threat opposes depends on the number of malicious users. If the number of wrongdoers is low, the risk is not major. Instead, if an attacked could somehow manage to get the majority of users (51%), this might become more problematic. More information on this concern will be offered in the next section.

5.1.2 51% Attack

As it is with most, if not all, the blockchain networks, the 51% attack represents a risk. If a group of attackers manage to obtain 51% of the total users in the application, their request answering capability rises. At this moment, monitoring the mining pool and the users in the application is enough to bypass this issue, but, in the future, some improvements like changing the consensus protocol from Proof-of-Work or making the mining process harder might come in handy [7].

5.1.3 Phising

Phishing attacks are one of the most common types of attacks. The only way to prevent this type of attacks, from the perspective of a solution developer, is to keep the users aware of the risks by conducting informing campaigns. They have to be careful when browsing the Internet and they have to keep their credentials and blockchain addresses to themselves [8].

5.1.4 Man in The Middle

Data that goes to MetaMask and to the Server Module is encrypted through HTTPS, thus making Man in The Middle attacks unfeasible for the communication with the Blockchain Module and the Server Module [9]. Even if an attacker could intercept data that is sent between the modules, the events that they could tamper or fraudulently introduce, would shortly be invalidated by the other traffic participants through the answering system.

5.2 Blockchain Performance

5.2.1 Storage Information

The number of blocks generated to hold all the transactions was 60, which, in terms of data storage, equals approximately 1.75 MB. It has been observed that one transaction generates a new block, meaning that a new block would occupy 30 KB. The 60 transactions are calculated as 30 introduced events and 30 deleted events for reevaluation, which means that, on average, 2 transactions are needed for every event.

Figure 10 shows the dependency between blocks in the blockchain and traffic events.

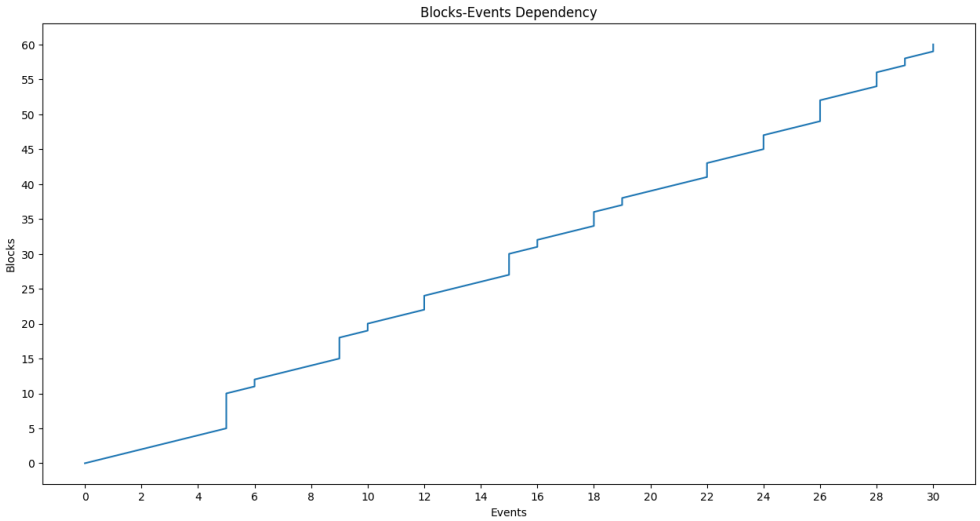

Figure 10: Blocks per Event Dependency

Compared to other blockchain solutions, like Bitcoin[1], that currently holds about 400GB of data[2], the records stored in the blockchain for this solution are expected to be significantly smaller in size and number, due to the fact that events can only be introduced at a fair distance from each other and that requests are subjective to users' confirmation.

5.2.2 Gas Fees

Figure 11 shows the gas fees for each transaction, during the experiment. The most important facts that are pointed out by the Figure are:

- The minimum value is 0.000745, highlighted by the green bar
- The maximum value is 0.004368, highlighted by the red bar
- The average value is 0.002767, highlighted by the blue horizontal line

[1] `https://bitcoin.org/en/`
[2] `https://ycharts.com/indicators/reports/bitcoin_statistics`

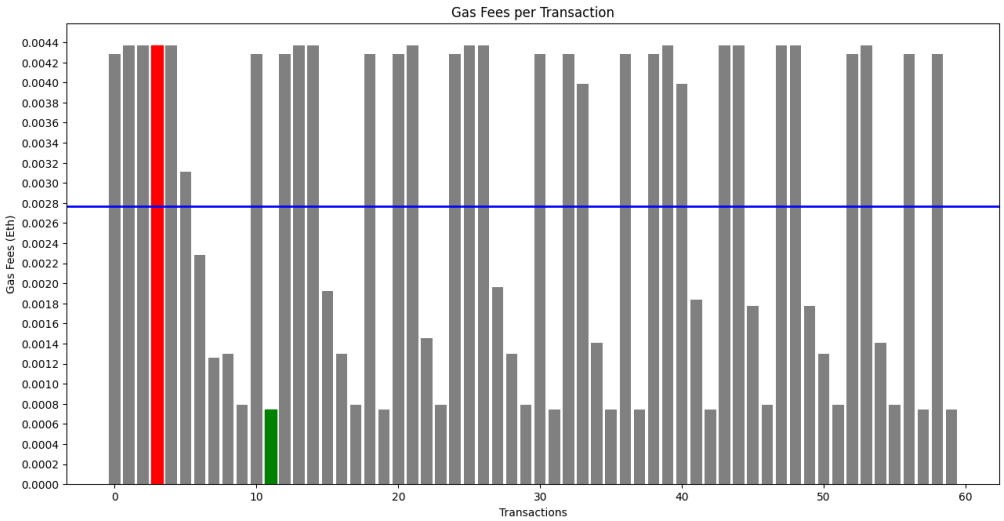

Figure 11: Gas Fees

5.2.3 Time Complexity

The time to fulfill a transaction adding process or a block mining process is nearly instantaneous, due to the performance of the blockchain chosen and also to the relatively small data stored in the blocks.

5.3 Application Performance

The code complexity of this solution is low, making the performance a core strength. Actions in the Client Module happen almost instantaneously. The Server Module, being developed in Javascript, NodeJS and ExpressJS, has really good performances and handles HTTP Requests really well due to its asynchronous processing. The single-threaded capacity of Javascript makes the implementation unable to be parallelized, but the native features of the technology, like event loop and the previously mentioned asynchronous processing allows the module to handle the requirements in a great way.

MongoDB is great in terms of performance, its pros being: high scalability, speed and high availability, the only mentionable downside being that it needs large storage capacity.

6 CONCLUSION

This paper proposed an improved way of using navigation applications, by keeping the involved users' confidentiality with blockchain and by making the traffic activity reliable due to the reputation each user has.

The solution accomplished to create a trustworthy, anonymous and credible environment for traffic participants. Every user that is using the application will only be identified by his randomly assigned address in the blockchain, therefore making him untraceable.

Any attempt of adding an event on the map has to be certified by the other users. The significance of any user's answer is being regulated by the reputation system. If a user with a high reputation answers to an event, his answer will value much more than the answer of a user with a low reputation. This assures that only the benevolent users will contribute to the network.

The implementation is fairly simple and self-explanatory, making this proposal a suitable solution for real-life scenarios. The main topics that future work could cover are:

- Building a blockchain network from scratch, one that suits the specific needs of this solution and that might have lower gas fees, better performance and security
- Choosing a different consensus protocol, in order to reduce the environmental impact that Proof-of-Work implicates and to maximize security

BIBLIOGRAPHY

[1] L. Hîrțan and C. Dobre. Blockchain privacy-preservation in intelligent transportation systems. *2018 IEEE International Conference on Computational Science and Engineering*, 2018.

[2] D. Cocîrlea, C. Dobre, L. Hîrțan, and R. Purnichescu-Purtan. Blockchain in intelligent transportation systems. *Electronics 2020*, 9(1682), 2020.

[3] L. Hîrțan, C. Dobre, and H. González-Vélez. Blockchain-based reputation for intelligent transportation systems. *Sensors 2020*, 20:791–813, 2020.

[4] S. Johar, N. Ahmad, A. Durrani, and G. Ali. Proof of pseudonym: Blockchain based privacy preserving protocol for intelligent transport system. *IEEE Access 2021*, 9:163625–163639, 2021.

[5] A. Balasubramaniam, M. Gul, V. Menon, and A. Paul. Blockchain for intelligent transport system. *IETE Technical Review 2020*, 38:438–449, 2020.

[6] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang. An overview of blockchain technology: Architecture, consensus, and future trends. *2017 IEEE 6th International Congress on Big Data*, 2017.

[7] S. Sayeed and H. Marco-Gisbert. Assessing blockchain consensus and security mechanisms against the 51% attack. *Appl. Sci. 2019*, 9:1787–1804, 2019.

[8] I. Astrakhantseva1, R. Astrakhantsev, and A. Los. Cryptocurrency fraud schemes analysis. *SHS Web of Conferences 2021*, 106, 2021.

[9] P. Ekparinya, V. Gramoli, and G. Jourjon. Impact of man-in-the-middle attacks on ethereum. *2018 IEEE 37th Symposium on Reliable Distributed Systems (SRDS)*, 2018.