Decentralized Vehicle Management and Transactions Using the CarConfiguration Smart Contract on Ethereum Blockchain

Bernard Paul
Computer Department
F.C.R.I.T
Vashi, India

Suhani Bhuti
Computer Department
F.C.R.I.T
Vashi, India

Purvita Dhakad
Computer Department
F.C.R.I.T
Vashi, India

***Abstract*—This paper presents a Solidity-based smart contract, CarConfiguration, designed to manage car information and facilitate transactions on the Ethereum blockchain. The contract offers decentralized solutions for adding, updating, purchasing, and retrieving car details, aimed at streamlining automotive transactions. By leveraging blockchain's transparency, security, and immutability, the CarConfiguration contract eliminates intermediaries, reduces transaction costs, and provides a trusted framework for vehicle ownership verification and sales.**

***Index Terms*—Blockchain, Smart Contracts, Student Credentials, Ethereum, Solidity, Decentralization, Verification.**

I. Introduction

Blockchain technology is revolutionizing traditional industries by offering decentralized solutions that enhance transparency and security. The automotive industry, however, still relies heavily on intermediaries, manual processes, and centralized databases, which can complicate vehicle sales and ownership verification. The CarConfiguration smart contract seeks to harness blockchain technology to create a transparent, trusted, and efficient system for managing vehicle data and sales.

A. Problem statement

In traditional car transactions, intermediaries such as dealers or brokers play a significant role in connecting buyers and sellers, verifying ownership, and handling funds. These third-party services increase costs and may lead to delays. Furthermore, current systems for vehicle ownership records are prone to errors, fraud, and tampering. This paper proposes a blockchain-based solution to address these challenges by automating vehicle registration, ownership updates, and payment transfers.

B. Objective

The objective of the CarConfiguration contract is to create a decentralized registry for vehicles, allowing for secure, transparent, and efficient car sales without intermediaries. By providing functionalities to add, update, and purchase cars directly on the blockchain, the contract enables buyers and sellers to interact directly, promoting trust and simplifying transactions.

II. Background and Motivation

Blockchain's immutability, decentralization, and security make it an ideal platform for applications requiring data integrity and transparency. However, the automotive industry has been slow to adopt blockchain-based systems for sales and ownership management. Implementing a blockchain solution can alleviate some key issues in traditional car transactions, including fraud, high transaction fees, and inefficient record-keeping.

A. *Current Challenges in Car Transactions*

1. Intermediary Dependence: Current systems heavily rely on intermediaries like car dealerships and brokers, which results in higher costs and can lead to trust issues.
2. Record Tampering: Centralized vehicle records are vulnerable to tampering or unauthorized modifications, potentially leading to fraudulent sales.
3. High Transaction Costs: Intermediaries add costs to car transactions, which could be reduced by implementing a blockchain-based solution.
4. Inefficient Ownership Transfer: Ownership transfer requires manual documentation and verification, leading to time-consuming procedures.

B. *Motivation for a Blockchain-Based Solution*

By implementing a decentralized smart contract, car transactions can become more secure, transparent, and efficient. The CarConfiguration smart contract manages vehicle records and ownership transfer autonomously, ensuring that data cannot be tampered with and funds are directly transferred between buyer and seller.

C. *The CarConfiguration Smart Contract*

The CarConfiguration contract is written in Solidity and deployed on the Ethereum blockchain. It has several main components that define its structure, including the data model, events, functions for adding, updating, and purchasing cars, and owner verification.

1. Car Structure and Data Model

The `Car` structure represents a car entity within the contract, containing essential details:

Make and Model: Brand and model of the car (e.g., Toyota Corolla).
Year and Price: Manufacturing year and price in Wei (the smallest unit of Ether).
Color: Car color as a string.
Engine and Fuel Type: Specifications for the car's engine and fuel type.

Availability: Boolean to track whether a car is available for purchase.

2. *Mapping for Ownership*

A mapping associates each car's ID with the owner's address. This ensures that only the legitimate owner can update or sell the car.

3. *Events*

Events are emitted for every key action in the contract, such as when a car is added, updated, or purchased. These events log details on the blockchain for transparency and allow clients to listen for contract changes.

III. Verification Methods

The contract includes several verification steps to secure transactions and ensure proper ownership:

A. *Ownership Verification*

The mapping of `carOwners` associates each car with its owner. Only the owner of a specific car can update its details or initiate its sale.
Attempts by unauthorized addresses to modify a car's data are restricted by requiring that the caller's address matches the owner's address.

B. *Fund Verification*

During a purchase, the contract verifies that the buyer has sent enough Ether to cover the car's listed price. If the sent Ether is less than the car's price, the transaction fails.
This check helps prevent underfunded purchases and ensures fair transactions.

C. *Availability Check*

Before completing a sale, the contract checks whether the car is still available. Once a car is sold, it is marked as unavailable to prevent double-selling.

IV. Implementation and Deployment

A. *Solidity Development Environment*

Developers need a Solidity-compatible development environment like Truffle or Hardhat, along with an Ethereum wallet such as MetaMask for deploying and testing the contract.
A test network, such as Ropsten or Rinkeby, can be used for initial testing before deploying to the Ethereum mainnet.

B. *Contract Compilation and Deployment*

The contract is compiled and then deployed to an Ethereum network, obtaining a unique address that users can interact with.
Deployment is done through tools such as Truffle or Hardhat, which simplify compiling and deploying contracts to Ethereum-compatible networks.

C. *Frontend Integration*

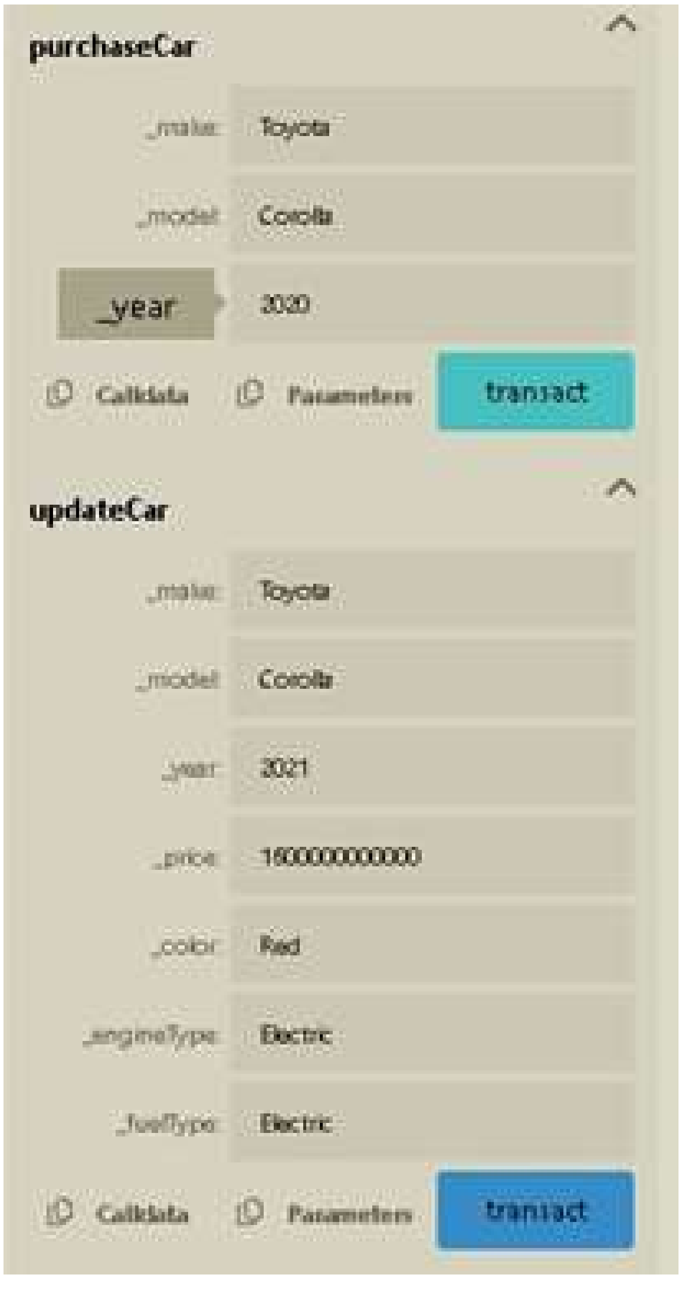
Frontend applications can be developed using libraries like Web3.js or Ethers.js to create a user-friendly interface that interacts with the contract.
This interface allows users to add cars, view car details, purchase vehicles, and track transaction history.

D. Testing

Key functions such as `addCar`, `updateCar`, `purchaseCar`, `getCar`, and `getTotalCars` are rigorously tested to ensure they work as expected.
Tests cover scenarios like insufficient funds, unauthorized updates, and availability status checks.

V. Results and Discussion

The CarConfiguration contract successfully demonstrates how decentralized vehicle transactions can operate securely and transparently

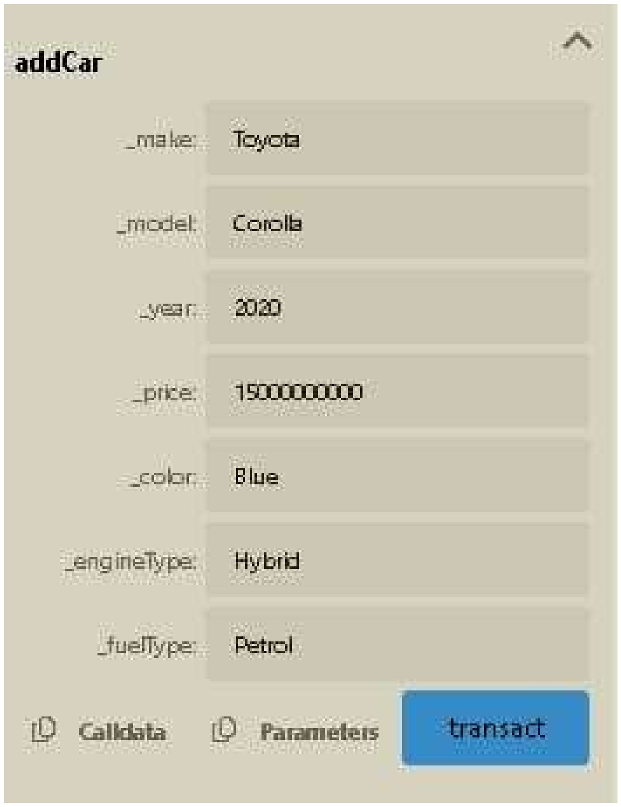


purchaseCar

_make: Toyota
_model: Corolla
_year: 2020

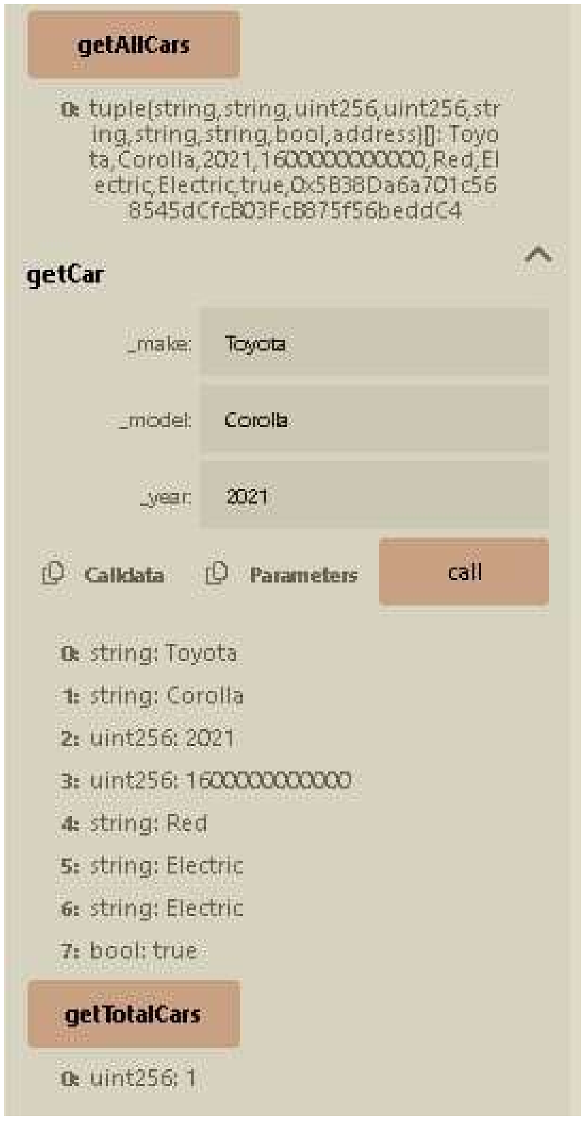
updateCar

_make: Toyota
_model: Corolla
_year: 2021
_price: 1600000000000
_color: Red
_engineType: Electric
_fuelType: Electric



addCar

_make: Toyota
_model: Corolla
_year: 2020
_price: 15000000000
_color: Blue
_engineType: Hybrid
_fuelType: Petrol



```
0: tuple(string,string,uint256,uint256,string,string,string,bool,address)[]: Toyota,Corolla,2021,1600000000000,Red,Electric,Electric,true,0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
```

getCar

_make: Toyota
_model: Corolla
_year: 2021

```
0: string: Toyota
1: string: Corolla
2: uint256: 2021
3: uint256: 1600000000000
4: string: Red
5: string: Electric
6: string: Electric
7: bool: true
```

```
0: uint256: 1
```

A. Key Achievements

Security and Transparency: The contract provides a secure environment for transactions, with each action recorded on the blockchain for transparency.
Elimination of Intermediaries: By allowing buyers and sellers to interact directly, the contract reduces dependency on intermediaries.
Ownership Protection: Only authorized owners can update or sell a vehicle, preventing unauthorized modifications and enhancing security.

B. Challenges and Limitations

Gas Costs: Each transaction (adding, updating, or purchasing a car) incurs gas fees. High gas fees can discourage users from interacting with the contract.
Scalability Concerns: As the number of cars and users grows, the contract may require optimization to handle increased storage and transaction loads.
User Adoption: While the blockchain approach offers security and transparency, it may require user education and interface improvements for mainstream adoption.

C. Potential Improvements

Optimizing Gas Efficiency: Simplifying data structures or using more efficient storage methods could reduce gas costs.
Layer 2 Solutions: Exploring layer 2 solutions (e.g., Optimistic Rollups) could reduce transaction costs and improve scalability.
Advanced Ownership Transfer: Implementing multi-signature approval or other mechanisms could add more robust ownership verification for complex transactions.

VI. Conclusion

The CarConfiguration contract demonstrates blockchain's potential to enhance vehicle transaction processes. By decentralizing the management of vehicle information and automating ownership verification and fund transfers, it offers a transparent and secure platform for car transactions. However, gas costs and scalability remain challenges for widespread adoption. Further optimization and integration with layer 2 solutions may enable this contract to serve as a viable model for blockchain-based automotive transactions in the future.

VII. ACKNOWLEDGMENT

The development of this contract was made possible by resources from the Ethereum community, along with open-source contributions from Solidity, Truffle, Hardhat, and Web3.js. Their extensive documentation, examples, and support forums were invaluable throughout the development and testing process. Special thanks to the blockchain developer community for their continuous innovation and contributions.

VIII. References

1. A. M. Antonopoulos and G. Wood, *Mastering Ethereum: Building Smart Contracts and DApps*, 1st ed. Sebastopol, CA: O'Reilly Media, 2018.
2. K. Christidis and M. Devetsikiotis, "Blockchains and Smart Contracts for the Internet of Things," *IEEE Access*, vol. 4, pp. 2292-2303, 2016. doi: 10.1109/ACCESS.2016.2566339.
3. M. Swan, *Blockchain: Blueprint for a New Economy*, 1st ed. Sebastopol, CA: O'Reilly Media, 2015.

4. V. Buterin, "A Next-Generation Smart Contract and Decentralized Application Platform," Ethereum White Paper, 2014. [Online]. Available: https://ethereum.org/en/whitepaper/.
5. A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts," in *2016 IEEE Symposium on Security and Privacy (SP)*, San Jose, CA, 2016, pp. 839-858. doi: 10.1109/SP.2016.55.
6. N. Szabo, "Formalizing and Securing Relationships on Public Networks," *First Monday*, vol. 2, no. 9, Sep. 1997. [Online]. Available: https://firstmonday.org/ojs/index.php/fm/article/view/548.
7. G. Wood, "Ethereum: A Secure Decentralized Generalized Transaction Ledger," Ethereum Yellow Paper, 2014. [Online]. Available: https://ethereum.github.io/yellowpaper/paper.pdf.
8. L. Xu, L. Chen, Z. Gao, Y. Chang, W. Li, and Y. Chen, "The Blockchain as a Platform for Smart Contracts Applications," in *2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, Exeter, 2017, pp. 1173-1178. doi: 10.1109/iThings-GreenCom-CPSCom-SmartData.2017.176.
9. I. Bashir, *Mastering Blockchain: Unlocking the Power of Cryptocurrencies, Smart Contracts, and Decentralized Applications*, 2nd ed. Birmingham, U.K.: Packt Publishing, 2018.
10. K. Z. Li and J. C. S. Lui, "Efficient Digital Contract Design for Fair Exchanges," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 6, pp. 1333-1345, Jun. 2018. doi: 10.1109/TIFS.2018.2795584.
11. A. R. Berente, S. Seidel, and R. Alt, "Smart Contracts in Asset Management: The Promise of Blockchain," *IEEE Computer*, vol. 51, no. 7, pp. 68-75, 2018. doi: 10.1109/MC.2018.3011045.
12. Y. Zhang and J. Wen, "The IoT Electric Business Model: Using Blockchain Technology for the Internet of Things," *Peer-to-Peer Networking and Applications*, vol. 10, no. 4, pp. 983–994, Jul. 2017. doi: 10.1007/s12083-016-0456-1