# Hotel Reservation with Smart Contract

Galura, Jarell Carlo, Garcia, Charles Albert,

and Tiamzon, Ralph Miguel

Holy Angel University

## I. Introduction

*Overview of the project.*

The use of blockchain technology has become increasingly popular in recent years due to its potential to improve security and transparency in various industries. One such application is the Hotel Reservation with Smart Contract project, which aims to leverage blockchain technology to streamline the hotel reservation process. In the traditional system, customers typically make a reservation by contacting the hotel directly or through a third-party booking platform. The reservation process typically involves filling out a form with personal information, selecting the desired dates and room type, and providing payment information. After the reservation is made, the hotel would then have to manually process the booking, which involves verifying the payment information and confirming the reservation details (Xu et al., 2020). This process can be time-consuming and prone to errors, which can lead to customer dissatisfaction and lost revenue for the hotel.

The Hotel Reservation with Smart Contract project utilizes blockchain technology and smart contracts to automate and secure the reservation process. Smart contracts are self-executing contracts that are stored on the blockchain and can be programmed to execute specific actions automatically when certain conditions are met. With this system, customers can make a reservation directly on the blockchain-based platform, which eliminates the need for third-party booking platforms and reduces transaction fees. The smart contract would then automatically verify the payment information and confirm the reservation details, which would help to eliminate errors and increase the efficiency of the booking process (Seyal & Ahmad, 2020).

The scope of the website will be to display the available hotel rooms and indicate whether they are currently open for booking. It will also facilitate the receipt of a security deposit and calculate the monthly rental fee. Furthermore, the staff or admin will have access to customer information and details of the room they have reserved.

*Explanation of the problem being solved.*

One of the primary reasons for using blockchain in this project is its ability to provide a secure and tamper-proof platform for storing and sharing data. With blockchain, all transactions are recorded on a decentralized ledger, which cannot be altered or deleted without the agreement of all parties involved (Drescher, 2019). This makes it much more difficult for hackers to compromise the system and steal sensitive information, such as credit card details.

Its ability to provide transparency and accountability makes both sides clear of any potential mismanagement of money and reservation. Since all transactions are recorded on the blockchain, it is possible to track every step of the reservation process, from booking to payment, in real time (Antonopoulos, 2014). This can help to prevent disputes between hotels and customers and ensure that everyone involved in the process is held accountable for their actions.

*Importance of Blockchain Decentralized Application in addressing the problem*

With regard to the issue raised earlier, this provides numerous advantages including improved efficiency, security, and transparency. On top of a blockchain network,

decentralized applications (dApps) are created with the goal of giving users a decentralized and trustworthy environment in which to interact (Swan, 2015).

By removing intermediaries, such as third-party booking platforms, and lowering the possibility of fraud or data manipulation, a blockchain-based dApp can offer a more secure and effective booking process in the context of hotel reservations (Seyal & Ahmad, 2020). Additionally, using smart contracts can automate and streamline the reservation process, lowering the risk of errors and speeding up transactions (Swan, 2015).

This approach can produce a more seamless and equitable booking experience for all parties involved by leveraging the advantages of blockchain and dApp technology.

## II. Design and Implementation

### *Overview of the Application*

Accessibility, security, and transparency are just a few advantages of using the web for a blockchain application project about hotel reservations (Munmun et al., 2021). The web-based blockchain-based hotel reservation system offers a user-friendly interface that is helpful for travelers who need to make reservations while on the go and can be easily accessed from any location with an internet connection. But because it is simpler to develop and use for a variety of platforms, the dApp developed for this project was a web.

The front-end of the application and the back-end of the application are the two main components of the dApp. As a web-based dApp, React.js was used to create the front end instead of traditional HTML, CSS, and JavaScript. According to the official React documentation, React is a declarative, effective, and flexible JavaScript library for building user interfaces, allowing you to compose complex UIs from discrete, small pieces of code known as "components" (React, n.d.). Moreover, React JS and Material UI are frequently combined in web development to create aesthetically pleasing and responsive user interfaces. Material UI's official documentation states that "Material-UI is a simple and customizable component library to build faster, beautiful, and more accessible React applications" (Material UI, n.d.). A variety of UI components are offered by Material UI, which can be easily modified to match the branding and design of a web application.

For the development of the back end, Hardhat was used. According to the official Hardhat documentation, it states that "Hardhat is a development environment to compile, deploy, test, and debug your Ethereum software. It helps developers manage and automate the recurring tasks that are inherent to the process of building smart contracts and dApps" (Hardhat, n.d.). The high-level programming language Solidity, which is used to create smart contracts for the Ethereum blockchain, is also supported by Hardhat. Hardhat "supports the Solidity language for smart contract development. It includes a Solidity compiler that can compile contracts to their EVM bytecode" (Hardhat, n.d.), which is stated in the official Hardhat documentation. Furthermore, Web3.js will be used to connect both the front-end and back-end. A connection between front-end and back-end applications and the Ethereum blockchain is made possible by the JavaScript library Web3.js. Through a web browser, it enables developers to communicate with smart contracts, send transactions, and read data from the Ethereum network (Ethereum, n.d.).

Additionally, developers can access blockchain functionality from within their applications by using Web3.js as an interface between the Ethereum blockchain and applications created on top of it.

*Explanation of the chosen Blockchain platform*

For this project, the chosen and used blockchain platform is Ethereum. Due to its adaptability and dependability, Ethereum is frequently selected as the blockchain platform for decentralized application (dApp) development. Buterin (2014) asserts that Ethereum is a decentralized platform that supports self-executing contracts known as smart contracts, in which the terms of the agreement between the buyer and seller are directly written into lines of code. Smart contracts are self-executing contracts. Without the use of middlemen, business transactions and processes can be automated using smart contracts. Ethereum is the ideal Blockchain to use in this case because the technology used to build the application, Solidity, Hardhat, and Web3.js, has been developed for or supports a great deal of the use of Ethereum. Furthermore, the majority of people can easily access Ether (ETH), the currency used to conduct transactions in Ethereum, through a variety of channels, and it is widely used. Ether is the second-largest cryptocurrency by market capitalization as of May 2023, according to CoinMarketCap (CoinMarketCap, 2023), and it is listed on hundreds of cryptocurrency exchanges around the world. According to GitHub, a popular platform for developers to share and collaborate on software projects, Ethereum has over 28,000 contributors and over 280,000 repositories as of May 2023 (GitHub, 2023). One of the benefits of the Ethereum ecosystem is its sizable and vibrant developer community, which fosters creativity and advances blockchain technology.

*Description of smart contract(s) used*

The website now utilizes a smart contract named HotelReservation, created using Solidity version 0.8.4. The file consists of 3 state variables:

- roomID
- price
- available

The roomID is the primary key to access all the functions in the Solidity file. RoomID is the unique ID of the room which is the basis of every function so that the functions can check the room status and if it exists. The price is the price of the Hotel Reservation fee. The available variable is a boolean to check if the said room is available or not by giving it true or false.

There are five (5) functions in the solidity file:

- addRoom()
- bookRoom()
- updateRoomAvailability()
- isRoomAvailable()
- rooms()

These five (5) functions all serve a purpose in the website. addRoom() function is to add the rooms that the user wanted to. The user will input the room ID and price of the room. bookRoom() function has the ability to book the room, however, the room must be added first through the addRoom() function. The updateRoomAvailability() has the ability to update the room from available or not available and vice-versa. In this case, the room must be added first. isRoomAvailable() function returns a boolean type so that if it's true, the room is

available. Otherwise, it's not. Provided that the roomID is already added. The last function of the file is the rooms(). This returns the availability of the room and the price.

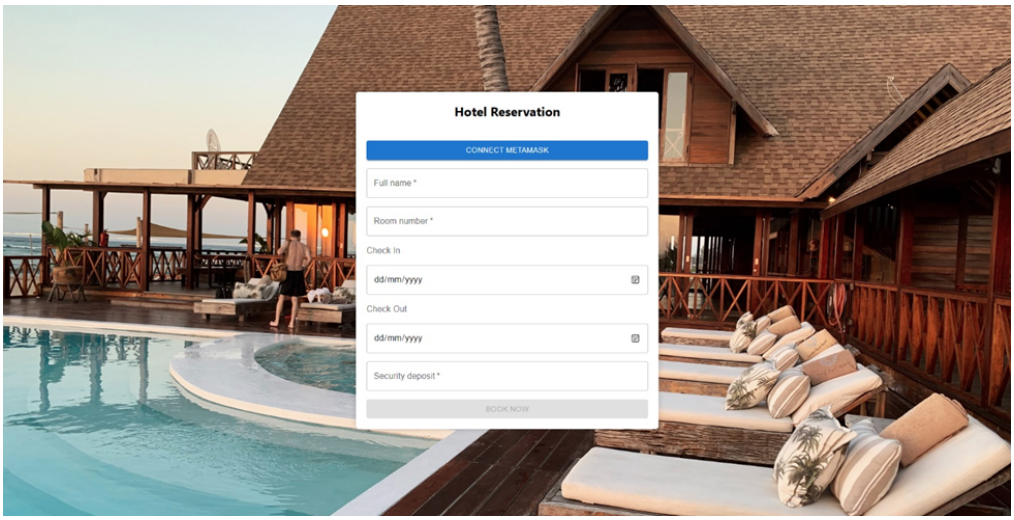
*Details of the decentralized application functionality*

The upgraded decentralized application will operate as a platform for booking hotel rooms. The website will provide information about the hotel, including its facilities, and feature a button for connecting your Metamask account. Upon connection, you will be redirected to the website. The website will check if the room you are interested in reserving is already occupied, and users can provide their personal details such as full name, room name, check-in date, and security deposit using forms. Once you book a room, the website will execute the bookRoom() function stored in the Solidity contract, which will accept the security deposit and verify its availability. If you decide to cancel your reservation, the website will display a separate dialog to confirm your cancellation. After booking a room, you will be redirected to a different page where you can view your reservation details, such as your full name, room name, check-in date, and security deposit information you provided earlier.

The dApp, on the front-end, consists of only one page. This page only consists of all the necessary steps for users to use which has one form and one button. This is the landing page (see Figure 1).

Figure 1

*Landing Page*




III. Testing

*Overview of the Testing Methodology*

Both the front end and the back end were tested. The front end of the dApp was tested in terms of checking the input fields, text fields, buttons, and any front-end things such as design, color scheme, and the layout of the forms. On the other hand, the back end of the dApp was tested manually by the use of Remix IDE to check for any deployment or syntax errors. This is to check the functions if they were functioning correctly.

*Explanation of the Test Cases Used*

The following test cases are the test cases referring to the correct function of the said smart contract functions:

Test Case 1: Test to see if the addRoom() function works correctly

Input: Number in Room ID and Number in Price

Expected Output: Successful adding of room

Test Case 2: Test to see if the addRoom() function works correctly

Input: Putting the same the Room ID but different Price

Expected Output: Message “Room already exists”

Test Case 3: Test to see if the addRoom() function works correctly

Input: Putting the same Price ID but different room ID

Expected Output: Successful adding of Room

Test Case 4: Test to see if the bookRoom() function works correctly

Input: Putting the Room ID that was already added

Expected Output: Successful booking of Room

Test Case 5: Test to see if the bookRoom() function works correctly

Input: Putting the Room ID that was not added

Expected Output: Message “Room not available”

Test Case 6: Test to see if the updateRoomAvailability() function works correctly

Input: Putting the Room ID and change it to available status

Expected Output: Successful change of status

Test Case 7: Test to see if the updateRoomAvailability () function works correctly

Input: Putting the Room ID and change it to not available status

Expected Output: Successful change of status

Test Case 8: Test to see if the isRoomAvailable () function works correctly

Input: Putting the Room ID that was added and booked

Expected Output: False

Test Case 9: Test to see if the isRoomAvailable() function works correctly

Input: Putting the Room ID that was not added and not booked

Expected Output: Message “Room does not exist”

Test Case 10: Test to see if the room() function works correctly

Input: Room ID

Expected Output: depends on the availability, either true or false and price

Figure 2

Test Case 1 Result

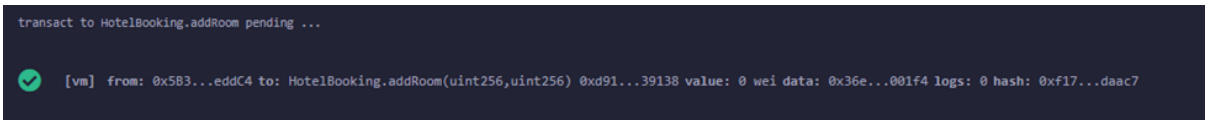



Figure 3

Test Case 2 Result

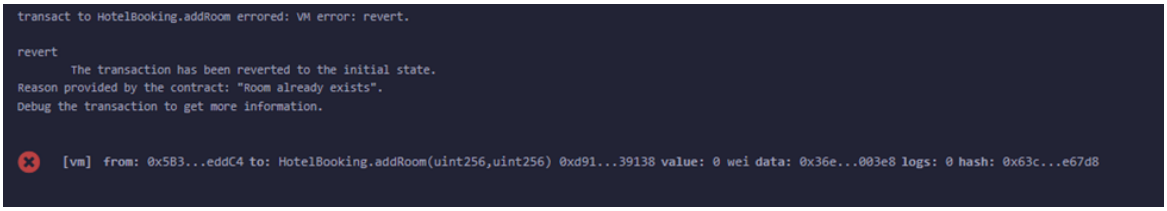



Figure 4

Test Case 3 Result

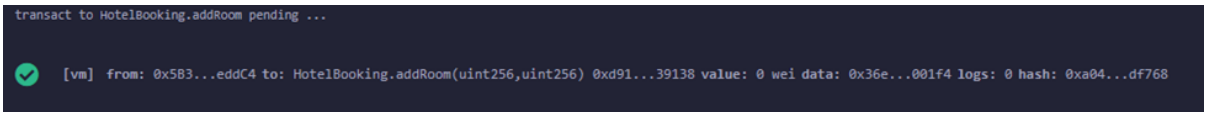



Figure 5

Test Case 4 Result

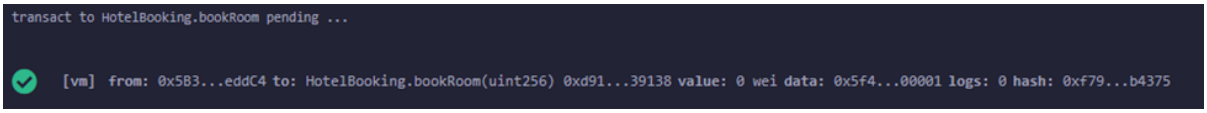
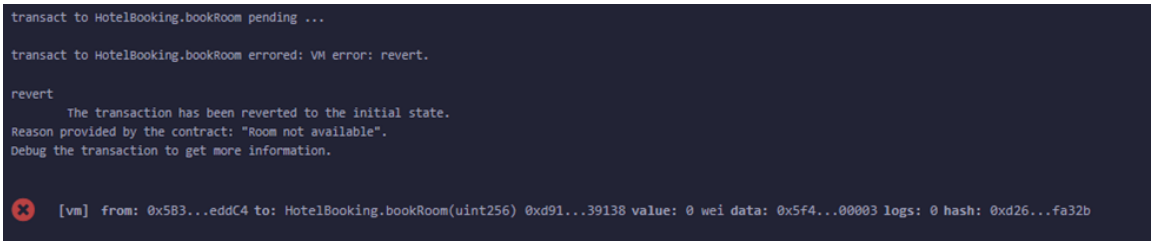



Figure 6

Test Case 5 Result




'Figure 7

Test Case 6 Result

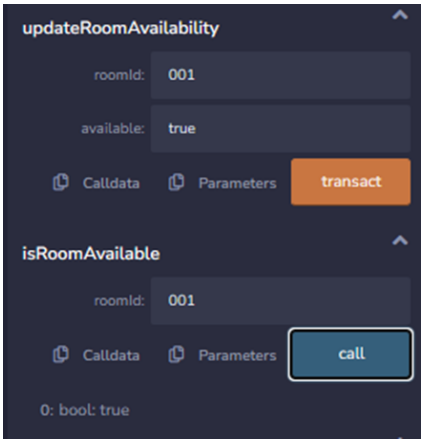




Figure 8

Test Case 7 Result

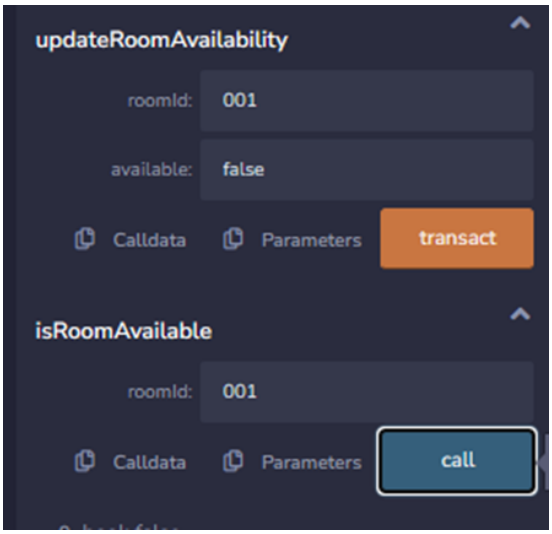




Figure 9

Test Case 8 Result


Figure 10

Test Case 9 Result

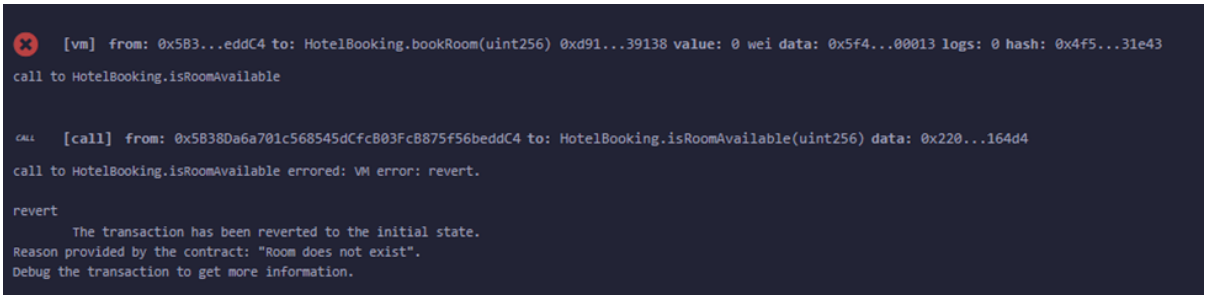
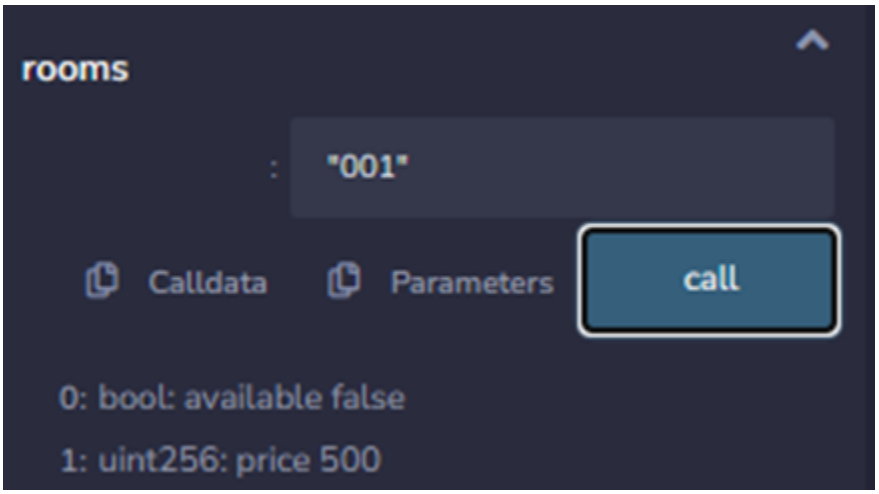



Figure 11

Test Case 10 Result




All the testing that was done indicate the results of the functionality of smart contract. All functions (addRoom, bookRoom, updateRoomAvailability, isRoomAvailable, rooms) works correctly as intended by the users.

IV. Deployment and Demonstration

*Description of the Deployment Process*

The dApp was hosted and deployed locally. The front end was deployed through the React.js project to its efficiency. By inputting the command "npm run start" on the visual studio code terminal, the front end of the application was deployed locally on localhost port 3000.

The back-end of the application or the Solidity smart contract was deployed using Hardhat. Ethereum network was created with 20 accounts of test wallet addresses through the command " npx hardhat node". These test wallets provided were provided by Hardhat.

*Overview of the Demonstration Environment*

To execute the Solidity contract, an Ethereum test environment was necessary for the software. Hardhat and its local node were used in this project to manage and launch the smart contract. To ensure the correct operation of the dApp, a library was needed to connect the smart contract with the application’s front end. In this project, Web3 was used as the library to facilitate communication between the Hardhat test environment that was created in the dApp. Specialized hardware was unnecessary for the hardware aspect. As the application was meant to be deployed locally.

*Explanation of the Demonstration Process*

The dApp can be accessed using a browser by entering the URL http://localhost:3000/ as if it were a webpage. Google Chrome was used to display the website for this demonstration. The dApp will route users to the landing page after they click the link. There will be a dialog box asking to install Metamask first if the browser does not already have any installed. Following the completion of that step, MetaMask will request authorization to link your given Hardhat test wallet account to the dApp. The users may now fill out the forms, select the dates they want to reserve, and the book now button will be enabled after they have verified the connection between the wallet and the website. After clicking the book now button, there will be a confirmation modal. As you click confirm in the modal, it will request the confirmation to the user's MetaMask in order to send the deposit of the hotel reservation fee.

*Details on how the decentralized application addresses the problem*

The issue of hotel reservations can be greatly improved by using decentralized applications. It can do away with middlemen like high-priced online travel agents and booking sites. These middlemen can be eliminated by implementing a decentralized application that makes use of blockchain and smart contracts. This enables direct communication between lodging establishments and guests, cutting expenses and raising transparency. Transparency and secure evaluations are further benefits. Reviews are essential in the decision-making process for booking hotels. Centralized systems, however, might be vulnerable to false or slanted evaluations. A decentralized application can offer transparent

and impenetrable review methods by utilizing blockchain technology. Users can post reviews that are authenticated and trustworthy by being permanently stored on the blockchain.

## V. Evaluation

### *Analysis of the Results of the Project*

The program was developed and included a Solidity smart contract that was implemented on a nearby Ethereum network. The study can be considered a genuinely decentralized application now that the smart contract has been successfully integrated and implemented within it.

The Hotel Reservation process was successful in fulfilling its purpose, which was to make it easy for you to pay the security deposit for the room you want to rent. Testing the application as a deployed service over the internet is one way that the project could be improved in the future. Upcoming projects comparable to this study could broaden their testing range to assess how the application responds to real-world usage.

### *Comparison with other solutions*

It was discovered that there is one comparison available that relates to the proposed solution; however, it is not accessible freely to the public. The comparison may have been conducted privately, or it may be part of a research study that has not been published or released yet. Unfortunately, without access to the comparison data, it is difficult to evaluate the solution's effectiveness or compare it to existing methods. Further investigation or

collaboration with the parties who conducted the comparison may be required to gain access to this information.

*Suggestions for Future Research*

Additional research in this area might investigate putting this solution to the test on a larger test population in the real world. While it was not specifically addressed in this project, future studies akin to it might also examine the level of familiarity that potential users currently have with blockchain technology as well as the rate of acceptance of such applications if they were to be put into practice in the real world. Furthermore, future researchers could concentrate on conducting user testing and gathering feedback to pinpoint areas that the website's forms and layouts that need to be improved upon. Research may also examine novel methods for creating simple, user-friendly interfaces that improve usability and engagement. This can aid future endeavors by predicting the likelihood of success and determining whether such applications can be accepted by the general user base.

*Explanation of the strengths and weaknesses of the decentralized application*

The advantages of adopting a decentralized application include the removal of intermediaries and trust and transparency, as was already mentioned. The removal of middlemen improves the efficiency of the reservation process because users may communicate directly with hotel providers, cutting out the middleman and associated costs. There are no middlemen or intermediates, so there is no centralized customer service to meet

the needs of the user, which is also a problem. Without a specialized customer care system, it could be harder to resolve any problems that arise throughout the reservation process.

## Conclusion

### *Summary of the Project*

An Ethereum-based decentralized application has been developed using React.js, Hardhat, Web3, and solidity. This was in response to the identified problems with the current solutions and based on the project's background. The Hotel Reservation process has effectively achieved its intended goal, which was to simplify the process of paying the room's security deposit for the accommodation you desire to book. The project has made clear both the potential benefits of such applications as well as any potential drawbacks that might carry difficulties in the deployment.

### *Significance of the Decentralized Application*

The created dApp firstly benefits customers as they gain the most from the dApp designed for hotel reservations because it enables them to successfully complete the transaction of booking a hotel room with confidence.

For this application, they also benefit and assist service providers in enhancing customer satisfaction and customer service. These blockchain-based applications are transparent by

nature, giving users access to valuable information that they can use to further refine their business models.

As for future researchers, this study is beneficial because it provides an example of a blockchain application that they can improve on. Furthermore, not only does it assist them in this area, it also helps to boost the popularity of decentralized applications because it can help inform people in general.

*Limitations of the Project*

One of the constraints on this project is that it has only undergone internal testing. It was not given the opportunity to log data that might have been obtained had it been tested by a larger population and set up across the internet instead of just a local test network.

Additionally, this project's limitations included the fact that the created application could only be used on computers with browsers that could work with the tools used, such as Hardhat, MetaMask, and similar ones.

Last but not least, the scope of this project was strictly limited to testing the crucial procedures, including reservations, getting and gathering payments, and information about payments.

*Suggestions for Future Research*

Additional research in this area might investigate putting this approach to the test on a larger number of individuals in a real-life setting. While it was not specifically addressed in this project, future studies akin to it might also examine the level of familiarity that potential users currently have with blockchain technology and the acceptance of such applications if they were to be applied in real life. This can aid subsequent projects by predicting the likelihood of their achievement and determining whether these kinds of apps can be accepted by the entire user base.

**References**

Antonopoulos, A. M. (2014). *Mastering Bitcoin: Unlocking Digital Cryptocurrencies. O'Reilly Media, Inc.*

Buterin, V. (2014). *A next-generation smart contract and decentralized application platform*. https://www.semanticscholar.org/paper/A-NEXT-GENERATION-SMART-CONTRACT-%26-DECENTRALIZED-Buterin/0dbb8a54ca5066b82fa086bbf5db4c54b947719a

Drescher, D. (2019). *Blockchain basics: A non-technical introduction in 25 steps. Apress*.

Ethereum. (n.d.). *Web3.js*. Retrieved from https://ethereum.org/web3-js/

Hardhat. (n.d.). *What is Hardhat?* Retrieved from https://hardhat.org/

Hardhat. (2021). *Hardhat Documentation*. Retrieved from https://hardhat.org/docs/

Lin, J., Xu, C., Chen, Y., & Xu, Z. (2018). *A Smart Contract-based Hotel Reservation System*. In Proceedings of the International Conference on Networking and Network Applications (NaNA'18) (pp. 149-153). ACM.

Munmun, S., Islam, R., Rahman, M. R., & Uddin, M. J. (2021). *Blockchain Technology and its Applications in the Hotel Industry*: *A Review*. Journal of Tourism and Hospitality Management, 9(1), 43-57.

Material UI. (n.d.). *Get Started*. Retrieved from https://mui.com/getting-started/overview/

React. (n.d.). *Introduction to React*. Retrieved from https://reactjs.org/docs/getting-started.html

Seyal, A. H., & Ahmad, N. (2020). *Blockchain technology: Opportunities and challenges in tourism and hospitality industry*. Journal of Hospitality and Tourism Technology, 11(4), 587-603.

Swan, M. (2015). *Blockchain: Blueprint for a new economy*. O'Reilly Media, Inc.

Xu, J., Zhang, Y., Ye, Q., & Law, R. (2020). *Understanding hotel booking behaviors on blockchain-based OTA*. International Journal of Contemporary Hospitality Management, 32(10), 3265-3282.