Decentralized hotel rooms booking system using Pragma solidity and blockchain technology

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering

By

AYUSH ANAND (Reg.No - 39110106)
AYUSH BHAGAT (Reg.No – 39110107)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE
JEPPIAAR NAGAR, RAJIV GANDHISALAI,
CHENNAI - 600119

APRIL - 2023

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Ayush Anand (39110106) and Ayush Bhagat (39110107)** who carried out the Project Phase-2 entitled **"Decentralized hotel rooms booking system using Pragma solidity and blockchain technology."** under my supervision from January 2022 to April 2023.

[illegible]

Internal Guide

Dr.S.PRINCE MARY, Ph.D

[illegible]

Head of the Department

Dr. L. LAKSHMANAN, M.E.,Ph.D.

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY ★ DEPARTMENT OF CSE

Submitted for Viva-voce Examination held on 20-04-2023

[illegible]

Internal Examiner

[illegible]

External Examiner

DECLARATION

I, **Ayush Anand (Reg.No- 39110106),** hereby declare that the Project Phase-2 Report entitled **“Decentralized hotel rooms booking system using Pragma solidity and blockchain technology”** done by me under the guidance of **Dr.S.Prince Mary, M.E, Ph.D** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering**.

[illegible]

DATE: 20-04-2023
PLACE:Chennai **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph. D**, **Dean**, School of Computing, **Dr. L. Lakshmanan M.E, Ph.D.,** Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.S.Prince Mary,M.E,Ph.D,** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my phase-2 project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

We introduce a decentralized platform for the home-sharing economy facilitating short and long-term stays payable with cryptocurrency and traditional payment methods. We are building a decentralized version of Airbnb . For sites like Airbnb, Blockchain is very useful. This is because it is capable of storing people’s online identities. This enables people to easily check if you are a trusted host by checking the ID number associated with your account. We are removing complete control of Airbnb We have built an Decentralized AirBnB that incorporates three main functionality ,Rent OUT your Space, View Available Space , Rent A Space From someone.

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS	DEFINITION
LOC	Lock Chain Tokens
OTA	Online Travel Agencies
DApp	Decentralized Application
ABI	Application Binary Interface
PoW	Proof-of-Work
PoS	Proof-of-Stake

CHAPTER 1

INTRODUCTION

In the recent COVID-19 time, The Hotel industry has immensely been affected, as we know that, Tourism is integrated with hospitality sector. After the covid, the hotel industry has grown significantly and show a massive growth. Thus, it needs to very transparent, while running. In terms of data, payments and all things, User has to stay in a comfortable zone. The integration of technology into the industry will surely be beneficial for the users as well as the business leaders.

Blockchain technology is one of the most exciting digital technology innovations of recent times, and despite its relative youth, it has the potential to fundamentally alter the way transactions are made and information is both stored and accessed. In this article, we take a closer look at the blockchain, explain what it is, and explore its current and potential future uses within the hospitality industry.

Blockchain technology has several advantages for those operating in hospitality management, with one of the most obvious being the security and stability benefits. For instance, all data is decentralised and traceable, and the database can never go offline, or be removed through a cyber-attack, which can be important when dealing with financial transactions.

In addition, technology could have a vital role to play in simplifying actual payments. At present, this can be somewhat complicated, especially when dealing with overseas settlements. With the use of blockchain technology, the entire process can potentially be streamlined and made more transparent, increasing trust.

Furthermore, the travel industry relies upon information and even personal possessions being passed between multiple different companies. The blockchain can make both accessing and storing information much easier, allowing for enhanced collaboration and ultimately improving the overall travel experience for customers.

Blockchain technology is very beneficial for hotelier because it has strong security and stability. Using this technology, the process of hotel management is made very easy and transparent. Customer can directly access the best hotel rates without any middleman so it will increase the customer experience.

Block Chain Hotel Management is Blockchain technology which eliminates the middleman and directly connects travel suppliers with tourists so travel suppliers can distribute their inventory without any intermediary’s fees and tourists can easily access their inventory without any markup fees on an exclusive platform.

It will decentralization hotel booking process and provide easy flow and allow hotel directly connect with their client. It enables sellers to get direct payments. The blockchain hotel management eliminates third-party costs and lays importance on direct seller to consumer interaction.

Blockchain Hotel Management potentially reduces market share of OTAs and intermediaries while making travel booking more affordable for the travellers. It is greatly beneficial in reservation systems, payment processing, and data sharing. Also, it ensures complete payment to all parties. It allows hoteliers to present and promote their available inventory to customers.

Blockchain Hotel Management Software allows direct transactions between travel companies and customers, enabling real-time transactions with no middleman fees. It helps hoteliers and hospitality companies manage hotel room availability and room rates in real-time and simplify hotel management processes.

Techno heaven introduces Blockchain Hotel Booking Technology for worldwide hoteliers to help them to rent their property at global level, collect money and manage bookings. This Blockchain Hotel Booking Technology plays a major role in eliminating the middleman which in turn helps the hoteliers in avoiding the unnecessary expenses of the middlemen.

In the recent times, Techno heaven has developed a Blockchain-based hotel distribution solution for the hotel owners. We are now connecting travel suppliers directly with tourists with the help of Blockchain Hotel Booking and we aim to make travel profitable for suppliers and cheaper for customers. Blockchain Hotel Booking Technology also allows all the enterprises to compete as small companies and large companies are brought on a same field.

The Blockchain Hotel Booking Technology is beneficial for travel agencies or travel agents in case of payment settlement and fraud prevention. Techno heaven Blockchain Hotel Booking Technology helps in decentralization hotel booking process and conveys more prominent straightforwardness by enabling hotels to directly interact with their clients.

Blockchain Hotel Booking Technology also creates a direct link between Hoteliers and its customer /clients. This Technology also enables direct access to the users so that they can access the best unbeatable hotel rates directly without any interference of the middlemen. Blockchain Hotel Booking Technology makes the management of inventories simpler and easier. Besides this the hoteliers can also provide more transparent services to their clientele.

With the help of Blockchain Hotel Booking Technology the suppliers can distribute their inventory directly to customers without any intermediaries' fees and the seller of tours or travels can access their inventory on an exclusive platform without any markup fees. With the use of Blockchain Hotel Booking the travel transactions can be managed easily so that the travel agents can concentrate more on other aspects of planning a trip.

Techno heaven Blockchain Hotel Booking Technology creates a decentralized database application for travel agents. Then hotel suppliers can register with our Blockchain Hotel Booking Technology and create e-vouchers and send a campaign to their agents, sell and manage easily with QR code control. It also lets you sell your hotel voucher unlimited and at global level. Authorized travel agents can browse and book group booking online easily through our hotel website and get instant confirmation.

This Blockchain Hotel Booking Technology enables hotels to concentrate more assets on giving better natural esteem and administrations to travellers around the world so it can reduce the financial point of hotels to third party booking platform.

In addition to lowering costs, booking through Lock Chain's marketplace is also more secure, as the blockchain technology is decentralized and transparent. Instead of credit cards, business travellers pay for hotels in Lock Chain tokens (LOC). All transactions are tracked via the LOC Ledger, a database that keeps track of all payment details. Typical hotel transactions outside of the LOC marketplace involve sharing credit card information through a vulnerable channel. There have been numerous examples of hacking of credit card data from poor digital infrastructure in the hotel. However, the LOC Ledger is decentralized, storing protected copies of the transaction in many places to prevent fraud. Transactions are time-stamped, secured through cryptography and added to the chain in a visible and reflexive way so that any tampering would be detected. This chain protects you against pesky credit card scams and sketchy hidden fees.

Tourism and blockchain have the potential to become a powerful combination as the technology can bring safety and transparency to several critical touchpoints. In the case of a travel agency booking flights and hotels for a customer, it has to send the information to the different firms. Blockchain could make this operation more secure and transparent since the responsibility spreads throughout the whole network. The same will happen with foreign transactions, increasing the level of trust among all involved parties.

4 Ways the Blockchain Can Be Used Within the Hospitality Industry

Although the potential uses for the blockchain within the travel industry are almost limitless, some applications have already emerged and are having a transformative effect, while others are just around the corner. Three of the most important ways the blockchain can be used are explained in more depth below:

1. Secure Payments: -

By far the most obvious practical application of the blockchain in the hotel industry will be in relation to secure payments. Whether it is the acceptance of cryptocurrency like Bitcoin or Ethereum, or simply the provision of a transparent, safe, global ledger, bank payments can be streamlined and associated costs for hotels can be reduced.

2. ID and Security: -

One of the most exciting potentials uses for the blockchain within the hospitality industry is related to identification and security services. Passengers are required to provide ID at various stages of their journey, but industry-wide adoption of the blockchain could potentially allow for a shared digital database, with passengers providing, for example, a finger print to quickly and seamlessly verify who they are, reducing waiting times.

3. Loyalty Programs: -

Loyalty reward schemes are an important part of generating return custom and blockchain technology can enhance the quality of loyalty programs by simplifying the process, making it easier for customers to access their points and redeem them. With the blockchain, rewards can be distributed through digital tokens that could, potentially, be used anywhere, at any time, while the inherent security benefits could reduce loyalty scheme fraud.

4. Baggage Tracking: -

Finally, the uses of the blockchain within the travel industry extend to things like luggage tracking. Typically, a traveller’s luggage changes hands many times over the course of a journey, which can result in logistical issues. The centralised nature of the blockchain can allow different companies to easily share and access related data.

CHAPTER 2

LITERATURE SURVEY

[1] A . Praveen et al., "Conference Room Booking Application using Flutter," 2020 International Conference on Communication and Signal Processing (ICCSP), 2020

Since there is a lot of ambiguity while booking a room at the last moment due to which we usually don't end up getting one. To avoid the occurrence of such a situation “Book The Room” app comes to your rescue. Through this app, [1] we can book the conference room well in advance based on the availability of the date and time that we have opted for. If the time limit exceeds a time span of 2 hours, then a request will be sent to the admin for necessary approval. Further to the approval, the conference room can be used. The queued requests are served on a first come first serve basis by the admin.

[2] H. Singh and R. R. Shah, "BOOKiiIT - Designing a Venue Booking System (Technical Demo)," 2020 IEEE Sixth International Conference on Multimedia Big Data (BigMM), 2020

Every academic institution has its system of managing the bookings of spaces. In an academic institution, rooms in different buildings may be handled by separate departments for booking purposes and most are carried out using the mail system. People using this system face numerous challenges like the absence of knowledge of available spaces, tediousness, improper management of bookings, clashes in bookings, communication failures, etc. These indicated a need for a well-defined, efficient, visualizable, and user-friendly space management system. BOOKiiIT, a Flutter app was proposed and designed using design thinking and PACT framework to make it user friendly and efficient in the different contexts of use. This app was implemented for the Indraprastha Institute of Information Technology (IIITD), but it can be easily replicated for other institutions.

[3] P. Somwong, S. Jaipoonpol, P. Champrasert and Y. Somchit, "Smart Room Vacancy Status Checking and Booking System," 2022 37th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC), 2022

In this work, the smart room system that checks the room vacancy in real-time is proposed. It also provides a web application for managing room reservations. To check the room vacancy in real-time, the loT devices collecting the temperature, the loudness of sound, and the light intensity are used to use as the datasets. These datasets are trained by machine learning to predict whether the room is vacant or not. The experiments are done to evaluate the performance of system prediction and the usage of the web application. The results show that the purposed system has a very high rate for both accuracy rate and Fl-score in room vacancy prediction. In addition, web application also satisfies users for managing the room schedule.

[4] S. Banerjee and A. Pal, "Luxury Hotel Booking and Scarcity Messages: Does Online Purchase Behavior Matter?," 2020 6th International Conference on Information Management (ICIM), 2020

Hotel booking websites commonly use scarcity messages to sell hotels' vacant room inventory. However, the effects of these messages on consumers' booking intention still remain unclear. Focusing specifically on luxury hotels, this paper seeks to address three research questions: (1) How do limited-quantity scarcity messages (e.g., “20% discount - Only 1 room left”) differ from limited-time scarcity messages (e.g., “20% discount- Only 1 day left”) in affecting consumers' luxury hotel booking intention? (2) How do frequent online purchasers differ from occasional online purchasers in their luxury hotel booking intention in response to scarcity messages? (3) Is there an interaction effect between scarcity message format (limited-quantity vs. limited-time) and online purchase frequency (frequent vs. occasional) on consumers' luxury hotel booking intention? Data came from 96 participants who took part in an online experiment. Results indicate that limited-time scarcity messages induced higher booking intention compared with limited-quantity scarcity messages. Moreover, frequent online purchasers exhibited higher booking intention compared with occasional purchasers. However, the interaction was non-significant. The findings have implications for luxury hotel managers, hotel booking websites, and online consumers. While frequent purchasers getting influenced by scarcity messages is a good sign for marketers, the paper serves to remind the consumers not to become shopaholic in terms of their online buying behavior.

[5] R. Dhanagopal, N. Archana and R. Menaka, "Enhanced Hotel Booking Application using PEGA," 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA), 2020

Flying colors is a Hotel Booking Application; this application provides different staying solutions to customers, it is to book customers a hotel room online based on date and location along with the type of hotel. This architecture is based on agile development which is specially designed to work on multiple regions, channels, and quick future enhancements. PEGA7 Business Process Management (BPM) Tool which enables significant enhancement in building the application and data management experiences mobile functionality, user experience, and analytics. The goal of developing this application using PEGA is to improve the efficiency and performance, the liveliness of usual processes in the business. An automated system is developed to deliver high security and end to end delivery of works and speeds up the process drastically.

[6] K. M. O. Nahar and R. M. Al-Khatib, "EPSSR: Energy preserving system for smart rooms," 2017 2nd International Conference on the Applications of Information Technology in Developing Renewable Energy Processes & Systems (IT-DREPS), 2017

Wastage of electricity is one of the main problems which we are facing now a day. In our home, school, colleges or industry we see that lights are kept on even if there is nobody in the room or area. This happens due to negligence or because we forgot to turn the lights off or when we are in hurry. In this paper, an Energy Preserving System for Smart Rooms (EPSSR) is proposed to save energy in smart rooms. Using the ESP8266 chip which is a Wi-Fi chip with full TCP/IP stack and MCU capability we developed a lighting controls to reduce electrical usage. Based on the technology of the Internet of Things (IoT), a lot of solutions may be done to control smart rooms light without the need of accessing the electrical sockets or plug. Our research idea focuses on measuring the number of persons entering any room like seminar hall, conference room and classroom using pair of Infrared sensors and the chip. When a person enters the room, counter will be incremented with lightening the room and the light continue lightening while persons counter greater than zero. When a person leaves the room, the counter is decreased by one. If the persons counter reaches zero, the lights inside the room will be turned off using a relay interface. This paper provides a real energy preserving model that could be used in daily life.

[7] L. M. Sánchez, I. Díaz-Oreiro, L. Quesada, L. A. Guerrero and G. López, "Smart Meeting Room Management System Based on Real-Time Occupancy," 2019 IV Jornadas Costarricenses de Investigación en Computación e Informática (JoCICI), 2019,

This paper proposes the creation of a smart meeting room through the incorporation of a PIR sensor and an AWS IoT button that allows the booking system to reflect a more precise availability of meeting rooms according to the actual occupancy status. The Internet of Things (IoT) devices are controlled using a Wi-Fi module that allows them to connect to the REST web service and to integrate with the open source Meeting Room Booking System (MRBS). In order to evaluate the system a storyboard evaluation was conducted with 47 participants. All participants filled out the User Experience Questionnaires (UEQ), described the product using three words and expressed their opinion through open comments. Finally, 19 participants took part in a real-life simulation of the smart meeting room and evaluated the system using the UEQ questionnaire. Based on the positive acceptance reflected in the evaluations, results show that the proposed system is considered very attractive and useful by the participants.

[8] L. Wu and Y. Wang, "A Low-Power Electric-Mechanical Driving Approach for True Occupancy Detection Using a Shuttered Passive Infrared Sensor," in IEEE Sensors Journal, vol. 19, no. 1, pp. 47-57, 1 Jan.1, 2019

Passive infrared (PIR) sensors are the most popular deployed sensors in buildings for individual presence detection. However, PIR sensors are motion detectors in nature, responding only to incident radiation variation, which leads to false negative detections, inaccurate occupancy estimation, and uncomfortable lighting swings and waste of energy. To address this issue, an optical shutter driven by a Lavet motor PIR (LAMPIR) sensor is developed for true presence detection. In comparison with our previously developed chopped PIR (C-PIR) and rotationally chopped PIR (Ro-PIR) sensors, a low-power single-phase electro-mechanical driving approach is introduced for LAMPIR to replace traditional servo and stepper motors and thus significantly reduce the power consumption by up to 89%, size by up to 60%, weight by up to 75%, cost by up to 31%, and acoustic noise by 12 dBA. More specifically, driven by pulsed signal from a microcontroller unit, the electro-mechanical vibrator drives a semi-transparent long-wave infrared optical shutter to chop the field of view (FOV) of a PIR sensor periodically. By monitoring and analyzing the voltage outputs generated by the LAMPIR senor, high-accuracy presence detection can be achieved by optimizing the shutter width and shuttering period through parametric studies. Experimental results show that a classification accuracy of 100% can be reached for detecting stationary occupants up to 4.5 m and moving occupants up to 10 m, suggesting a detection range improvement from both the C-PIR and the Ro-PIR sensors (4.0 m for stationary and 8.0 m for moving occupancy detection for both sensors). Additionally, the LAMPIR sensor has an FOV of 90° in horizontal and 100° in vertical, which is sufficient for most applications. For a 31-h-long presence detection test, an accuracy of 97% is achieved when classifying unoccupied and occupied scenarios, while an accuracy of 93% is achieved when classifying unoccupied, stationary and moving occupant scenarios.

[9] M. Saravanan and A. Das, "Smart real-time meeting room," 2017 IEEE Region 10 Symposium (TENSYMP), 2017

The monumental changes happening in present infrastructure industry can mostly be attributed to developments in Internet of Things. Developing smarter conference rooms in offices have a great scope for innovation and digital transformation. In this paper, we have developed a Smart Meeting Room (SMR) architecture and addressed various imminent issues pertaining to a meeting room environment by providing automation. The Smart Room requirements at real-time are automatically customized and act according to the customer's needs. Due to the developments in Machine to Machine (M2M) communication, we can establish a connection between the devices and the customers on a real-time basis. This enables the smart rooms to interact with the users and act according to their needs as well as automatically establish communication with other meeting rooms. Various types of sensors like temperature sensor, humidity sensor, PIR (Passive Infrared) based motion detector etc. are deployed in the smart room to monitor and enable the actuators. Ericsson's APPIoT framework facilitates smooth discovery, attachment and data sharing between devices in a close proximity. Gnokii and Pidgin systems are used with our architecture to ease the customization and execution of different events. A microcontroller serves as the brain and gateway for the control operations and provides users with additional features such as setting up of the room environment, loading presentations on the display in advance etc. By continuously learning from the environment within the room, the actuators trigger a responsive event automatically to control the room's ambience and the bookings

[10] L. Duc Tran et al., "A smart meeting room scheduling and management system with utilization control and ad-hoc support based on real-time occupancy detection," 2016 IEEE Sixth International Conference on Communications and Electronics (ICCE), 2016

In most meeting room scheduling or management system, the availability of meeting rooms are mainly based on pre-determined schedules. However, since the meeting duration is not always exact as it is scheduled, there are some situations that a meeting room is underutilized. Therefore, in this paper, we present a smart meeting room scheduling and management system which detect occupancy status of meeting rooms in realtime and integrate this information into the scheduling application to support ad-hoc meetings and increase room utilization. Our system is a simple, ease-of-implementation solution based on PIR sensor fusion devices and Ethernet connectivity. Occupancy data is sent to a central application server by UDP over IP protocols. On this server, a web application is developed and hosted to not only allow people book rooms for their meetings, but also check the utilization of these rooms based on predefined policies. The system also supports ad-hoc meetings by providing real-time availability of meeting rooms to users.

2.1 INFERENCES FROM LITERATURE SURVEY

From the above-mentioned literature works, it is clear that there has been effective research on this topic has been done and many models have been proposed. It is evident that the above-mentioned systems have their own pros and cons. While some of the recent works involve hybrid technologies and provide better accuracies, they are still far from what is needed. With higher accuracy, comes the need for low computational costs, high processing speed, and most of all, the convenience of use.

2.2 OPEN PROBLEMS IN EXISTING SYSTEM

In the existing system , the platform charge fees up to 20% of the total booking cost The existing system is implemented with less security mechanisms thus making the system prone to SQL injections and other data attacks The existing system has simple use case and cannot be scaled to complex use cases.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 AIM AND SCOPE OF THE PRESENT INVESTIGATION

The aim of this project is to develop a decentralized hotel room booking system using Pragma Solidity and blockchain technology. The primary goal of the system is to provide a secure and efficient way for customers to book hotel rooms without the need for intermediaries.

Currently, the hotel room booking industry is dominated by centralized platforms, which act as intermediaries between customers and hotels. These platforms charge high fees for their services, and customers are often left with little control over the booking process. Additionally, these centralized systems are vulnerable to security breaches, which can compromise user data and lead to fraud.

By leveraging blockchain technology and Pragma Solidity, we aim to create a decentralized system that eliminates the need for intermediaries and provides increased security and transparency. With this system, customers will be able to book hotel rooms directly with the hotel, using cryptocurrency as payment. This will reduce costs for both customers and hotels, while also increasing efficiency and transparency.

To achieve this aim, we will use Pragma Solidity to create a smart contract that will handle the booking process. The smart contract will be deployed on the blockchain, ensuring that all transactions are secure and transparent. Customers will be able to interact with the smart contract through a decentralized application or web interface, making the booking process easy and accessible.

The smart contract will contain all the necessary rules and regulations for booking a room, ensuring that the process is fair and transparent. When a customer wants to book a room, the smart contract will check the availability of the room and create a new booking if the room is available. Payment for the booking will be made using cryptocurrency, which will be transferred directly to the hotel's wallet address.

Once the booking is confirmed, the smart contract will send a notification to the customer and the hotel, confirming the details of the booking. The booking details will be stored on the blockchain, providing an immutable record of the transaction. This will ensure that all parties have a transparent and secure record of the booking process.

In conclusion, the aim of this project is to create a decentralized hotel room booking system using Pragma Solidity and blockchain technology. This system will provide increased security, transparency, and efficiency compared to traditional centralized systems, and will eliminate the need for intermediaries. By leveraging blockchain technology, we aim to create a more secure and efficient booking system that benefits both customers and hotels.

A decentralized hotel room booking system built using the Pragma Solidity language and blockchain technology would have several advantages over traditional centralized systems. Here are some of the key benefits:

1. Increased security: Blockchain technology ensures that all transactions are secure and tamper-proof, providing greater protection against fraud and hacking.
2. No intermediaries: Decentralized systems eliminate the need for intermediaries such as booking agents, reducing costs and increasing efficiency.
3. Transparency: Blockchain technology provides an immutable record of all transactions, ensuring transparency and accountability.
4. Lower costs: Decentralized systems are typically less expensive to operate than centralized systems, as there are no intermediaries to pay.
5. Faster processing: Blockchain technology allows for faster processing of transactions, as there are no intermediaries to slow down the process.

To build a decentralized hotel room booking system, you would need to start by creating a smart contract in Pragma Solidity that would handle the booking process. The smart contract would be deployed on the blockchain and would contain all the necessary rules and regulations for booking a room.

When a user wants to book a room, they would interact with the smart contract using a decentralized application (DApp) or web interface. The smart contract would check the availability of the room and, if it is available, create a new booking and record the details on the blockchain.

Payment for the booking would also be handled through the smart contract. Users would pay for their bookings using cryptocurrency, which would be transferred directly to the hotel's wallet address.

Once the booking is confirmed, the smart contract would send a notification to the user and the hotel, confirming the details of the booking. The booking details would be stored on the blockchain, providing an immutable record of the transaction. One potential benefit is the reduction of costs for both customers and hotels. By eliminating intermediaries and reducing transaction fees, the cost of booking a hotel room could be significantly reduced. This could help to make travel more accessible and affordable for a wider range of customers, while also improving the profitability of hotels by reducing their reliance on third-party booking platforms.

Another potential benefit is the increased security and transparency of the booking process. By using blockchain technology and smart contracts, we can ensure that all transactions are secure, transparent, and tamper-proof. This can help to reduce the risk of fraud, protect customer data, and increase trust between customers and hotels.

Additionally, a decentralized hotel room booking system could help to promote the use of cryptocurrency as a payment method for travel and hospitality services. Cryptocurrency payments can offer lower transaction fees, faster processing times, and increased security compared to traditional payment methods. By accepting cryptocurrency payments, hotels can reduce their dependence on traditional payment systems and offer a more convenient and secure payment option for customers.

Moreover, the development and implementation of a decentralized hotel room booking system could contribute to the wider adoption and development of decentralized systems and blockchain technology. By demonstrating the potential benefits of decentralized systems for the hospitality industry, we could encourage other hotels and hospitality providers to explore the use of blockchain technology for their own operations.

Finally, a decentralized hotel room booking system could help to promote innovation and competition in the hospitality industry. By reducing the barriers to entry for new players and enabling direct bookings between customers and hotels, a decentralized system could promote a more competitive and innovative industry that benefits both customers and hotels.

In conclusion, there are several potential benefits that could be realized through the development and implementation of a decentralized hotel room booking system using Pragma Solidity and blockchain technology. These benefits include the reduction of costs for both customers and hotels, increased security and transparency, promotion of cryptocurrency payments, contribution to the wider adoption and development of decentralized systems, and promotion of innovation and competition in the hospitality industry.

In summary, a decentralized hotel room booking system using Pragma Solidity and blockchain technology would provide increased security, transparency, and efficiency compared to traditional centralized systems. With the growing adoption of blockchain technology, decentralized systems are becoming more common, and it's likely that we will see more decentralized booking systems in the future.

3.2 FEASIBILITY STUDIES/RISK ANALYSIS OF THE PROJECT

FEASIBILITY STUDY

A decentralized application for Airbnb using blockchain technology is a feasible idea. Blockchain technology can offer several advantages for such a platform, such as a decentralized system that is less prone to fraud and data breaches, transparent transactions, secure data sharing, and peer-to-peer transactions without intermediaries. However, there are also some challenges to consider, such as scalability issues, the need for significant investment in infrastructure and development, and the need to comply with regulations in different countries and jurisdictions. Additionally, many people who use Airbnb may not be familiar with blockchain technology, which could be a barrier to adoption. Despite these challenges, several companies are already working on developing blockchain-based platforms for the travel industry, which could pave the way for a blockchain-based Airbnb in the future.

Three key considerations involved in the feasibility analysis are

- Risk Analysis
- Technical feasibility

Risk Aanalysis

Risk analysis is a process of identifying, assessing, and managing potential risks associated with a project or business. Here is an example of a risk analysis for a dApp like Airbnb using blockchain technology:

- Security Risks: The security of the dApp is a critical concern, as any breach could lead to theft of user information or assets. The risks could include hacking attempts, malware, or phishing attacks. Possible mitigations include encryption of user data, multi-factor authentication, regular security testing, and insurance against security breaches.

- Regulatory Risks: The regulatory environment for blockchain-based platforms is complex and constantly evolving, which creates uncertainty for developers and users alike. The risks could include changes in laws and regulations, lack of clarity around taxation, and potential legal challenges related to real estate or travel transactions. Possible mitigations include engaging legal counsel to ensure compliance, lobbying for favorable regulatory treatment, and developing alternative models that reduce regulatory risk.

- Operational Risks: The operation of a dApp like Airbnb would require specialized knowledge and skills, which may be in short supply. The risks could include downtime due to technical issues, high transaction fees, and difficulties in managing customer support. Possible mitigations include hiring experienced blockchain developers, providing training for staff, using reliable hosting and infrastructure, and developing a user-friendly interface.

- Financial Risks: Developing and operating a dApp like Airbnb would require significant financial investment, which creates a risk of financial loss if the platform fails to attract users or generate revenue.

TECHNICAL FEASIBILITY

Developing a decentralized application for Airbnb using blockchain technology is technically feasible, but there are some challenges that need to be addressed. These challenges include scalability issues, the need for complex smart contracts, ensuring the security and privacy of user data, and addressing the issue of user adoption. However, with careful planning and implementation, it is possible to overcome these challenges and create a successful blockchain-based platform for Airbnb. Several companies are already working on developing blockchain-based platforms for the travel industry, which indicates that it is possible to develop a blockchain-based Airbnb platform in the future..

3.3 SOFTWARE REQUIREMENTS SPECIFICATION DOCUMENT

Hardware specifications:

- Higher RAM, of about 4GB or above
- Frequency of 1.5GHz or above
- Storage of 32GB or above.

Software specifications:

- Python 3.6 and higher
- Hardhat
- Anaconda

CHAPTER 4

EXPERIMENTAL OR MATERIALS AND METHODS AND ALGORITHMS USED

4.1 SELECTED METHODOLOGY OR PROCESS MODEL

Description : We are building a decentralized hotel room booking system using Pragma Solidity and blockchain technology. This system aims to provide a secure and transparent way for hotels to manage their bookings, without the need for intermediaries like online travel agencies (OTAs) or centralized booking platforms. The system will allow customers to browse available rooms and make bookings directly on the blockchain, with the smart contract code ensuring that only valid bookings are accepted based on factors like room availability, customer identity, and payment confirmation. Payments for bookings will be held in escrow on the blockchain until the customer checks into the hotel and confirms that the room meets their expectations. Once the customer confirms the booking, the payment will be released to the hotel. Overall, this system will offer a more efficient and cost-effective solution for both hotels and customers in the hospitality industry.

We plan to build a Decentralized AirBnB that incorporates three main functionalities

- Rent OUT your Space
- View Available Space
- Rent A Space From someone

Full Stack DApp deployed on Polygon (Matic) with Proof of Stake Security. As a DApp developer, to build on PoS security, the procedure is as simple as taking your smart contract and deploying it on Polygon (Matic). This is possible because of the account-based architecture enabling an EVM-compatible sidechain.

Module 1 : Smart Contract Part Of Our Project :

In this module of our project we are writing blockchain logic or Blockchain logic of our project . The smart contract is a critical part of our decentralized hotel room booking system. The smart contract is a self-executing program that automatically enforces the rules and regulations of the booking process without any need for human intervention. It will ensure that only valid bookings are accepted, based on factors like room availability, customer identity, and payment confirmation.Order of contract files is super important when we are using testing tool like Hardhat . We are using Solidity for writing blockchain logic of our project. It contains components like :-

- rentProperty()-To make a booking
- markPropertyAsInactive() -To take down a property from the market
- sendFund() - for transferring the funds from one account to another.
- CreateBooking() - for create a booking of the property
- rentOutproperty() -To put up a property for rent on Airbnb market

Module 2 : Frontend Part Of Our Project :

We are using react.js to create frontend of our project . We will use web3.js to connect our frontend with smart co tract code through ABI of our contract. APIs facilitate the interaction between web applications and centralized servers, and the Solidity ABI provides smart contract data to applications and other contracts. When an application uses an API to request data from a server, the API feeds it, whereas ABIs access smart contract data in binary bytecode format known as Solidity Binaries.

Now the question is:

The Application Binary Interface (ABI) is an interpreter that facilitates communication with the EVM bytecode. The Solidity ABI is a human-readable list of methods on a smart contract for executing particular functions. You can use the ABI with a library like ethers.js to interact with smart contracts. ABI stands for Application Binary Interface. In the context of blockchain development, ABI refers to the interface that defines how software components interact with each other on the blockchain.

In Ethereum and other blockchain platforms that support smart contracts, the ABI is a standard way to describe the functions, arguments, and return values of a smart contract. ABI is essentially a blueprint for how a smart contract can be called, including the function signatures and the types of the arguments and return values.

The ABI is important because it allows different components of a blockchain ecosystem to communicate with each other in a standardized way. For example, a user interface application that interacts with a smart contract can use the ABI to generate function calls to the contract and interpret the return values. Similarly, a blockchain explorer tool can use the ABI to decode transaction data and display it in a human-readable format.

An ABI in Solidity is similar to, but also different from, an API (Application Program Interface).

In web2, APIs facilitate the interaction between web applications and centralized servers, and the Solidity ABI provides smart contract data to applications and other contracts. When an application uses an API to request data from a server, the API feeds it, whereas ABIs access smart contract data in binary bytecode format known as Solidity Binaries.

In summary, the ABI is an important aspect of blockchain development as it defines the interface between different components of a blockchain ecosystem, enabling seamless communication and interoperability between them.

Module 3: Connect UI to Metamask:

Metamask is a browser extension that serves as a wallet and gateway to the Ethereum blockchain. It allows users to manage their Ethereum accounts, sign transactions, and interact with decentralized applications (dApps) directly from their web browser.

With Metamask, users can create and import Ethereum accounts, view their account balance, and send and receive Ether and ERC-20 tokens. Metamask also provides a secure way to sign transactions and interact with dApps by injecting a

Web3.js instance into the web page, which enables the dApp to communicate with the Ethereum blockchain.

Metamask is available as a browser extension for Google Chrome, Firefox, Brave, and other Chromium-based browsers. It is a popular tool for developers and users of Ethereum dApps, as it provides a convenient and

secure way to interact with the Ethereum blockchain directly from the web browser.

we will connect the UI to metamask.for this we have to follow 2 steps, first add the setProvider() method inside mounted() , next we would like to inject the metamask instance for this we define setProvider() method.Metamask will be used which acts as a bridge between websites and Ethereum blockchain.The smart contracts in the network can access the data or not. All the UI part is implemented on react.js which is open JavaScript library for Building user interfaces from UI components.

4.2 ARCHITECTURE / OVERALL DESIGN OF PROPOSED SYSTEM

FLOW DIAGRAM

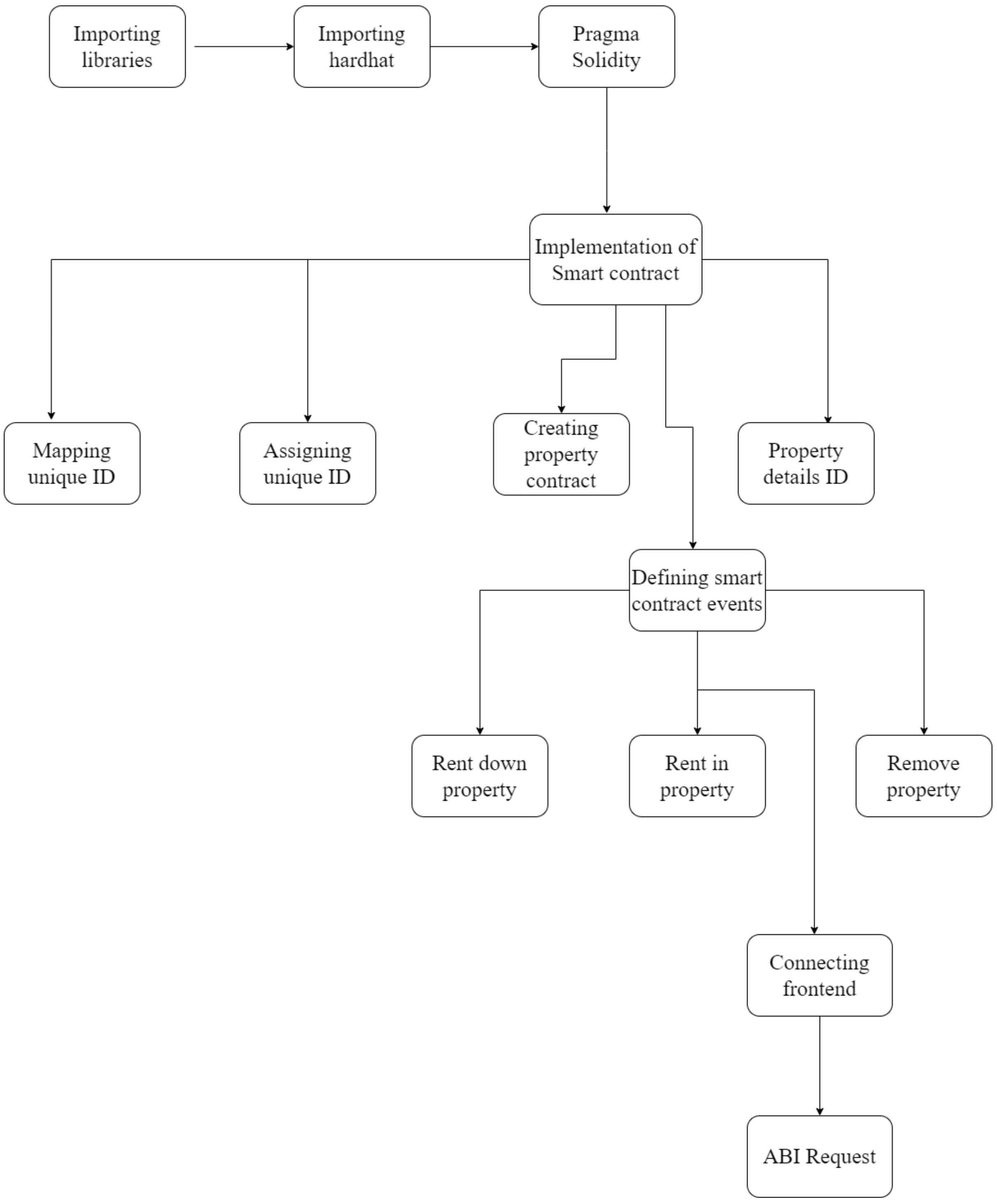


Fig 4.1 Flow Diagram of Decentralized Application and working

In figure 4.1 we can understand the working flow of the our decentralized hotel booking application , the internal working of the application.

•**Import libraries**-

In a DApp hotel booking application, you may need to use various libraries depending on the programming language you are using to build the application. Here are some commonly used libraries for building DApps:

• **Web3.js** - Web3.js is a powerful JavaScript library that enables developers to interact with Ethereum nodes using the JSON-RPC protocol. It provides a wide range of features for building decentralized applications, including querying blockchain data, creating and signing transactions, and interacting with smart contracts. Web3.js also allows for easy integration of Ethereum wallet functionality into DApps, making it simple for users to manage their accounts and interact with the blockchain. The library provides a straightforward API for interacting with smart contracts, including methods for reading data, executing transactions, and listening for events emitted by the contracts. With its broad range of capabilities and easy integration, Web3.js is an essential tool for any developer building a DApp.

• **Truffle** - Truffle is a popular development framework for Ethereum that provides developers with a suite of tools for building decentralized applications. It simplifies the process of developing, testing, and deploying smart contracts, and provides a range of utilities for interacting with the Ethereum blockchain.Truffle includes a built-in smart contract compiler, which compiles contracts written in Solidity, the most popular programming language for writing smart contracts on Ethereum. It also includes a testing framework that allows developers to write and run automated tests for their contracts.One of the key features of Truffle is its integration with the Ganache personal blockchain. Ganache provides a local blockchain environment for testing and development purposes, making it easy for developers to test their contracts and DApps without having to deploy them to the main Ethereum network.Truffle also includes a suite of tools for deploying smart contracts to the Ethereum network. It supports multiple deployment options, including deploying contracts to a local blockchain or to a public network like the Ethereum mainnet or a testnet.

Overall, Truffle provides a comprehensive set of tools for building Ethereum-based decentralized applications, and is widely used by developers in the Ethereum ecosystem.

- **Ganache** - Ganache is a personal blockchain for Ethereum development. It allows developers to test and deploy smart contracts, develop decentralized applications (DApps), and simulate different blockchain scenarios in a sandbox environment.Ganache comes in two versions: Ganache CLI and Ganache GUI. Ganache CLI is a command-line tool that provides a fast and configurable way to run an Ethereum blockchain locally. Ganache GUI is a user-friendly desktop application that provides a visual interface to interact with the blockchain.Some of the features of Ganache include the ability to create multiple accounts with different balances, simulate various network conditions, and interact with the blockchain through a built-in web3 provider. Ganache is widely used by developers for testing and debugging smart contracts before deploying them on the main Ethereum network.

- **Solidity** - Solidity is a high-level programming language used for developing smart contracts on the Ethereum blockchain platform. Smart contracts are self-executing contracts with the terms of the agreement between buyer and seller being directly written into lines of code.Solidity was developed by the Ethereum Foundation and is similar to JavaScript in syntax. It is designed to be secure, easy to read and write, and compatible with other programming languages such as C++, Python, and JavaScript. Smart contracts written in Solidity can be used for a wide range of applications, including decentralized finance (DeFi), supply chain management, identity verification, and more. Solidity provides a variety of features such as data types, control structures, functions, and libraries to help developers create efficient and secure smart contracts.To deploy smart contracts written in Solidity, developers use the Ethereum Virtual Machine (EVM) which executes the code on the Ethereum blockchain. The Solidity compiler is used to convert the Solidity code into EVM bytecode, which is then executed by the EVM.Overall, Solidity is an important tool for developers who want to build decentralized applications on the Ethereum blockchain.

- **OpenZeppelin -** OpenZeppelin is an open-source framework for building secure smart contracts on blockchain platforms such as Ethereum. It provides a library of reusable and audited smart contract components that developers can use to build decentralized applications (dApps).The framework includes modules for various functionalities such as token creation, access control, ownership, and more. It also provides tools for testing and deploying smart contracts.One of the key features of OpenZeppelin is its focus on security. The smart contracts provided by the framework have been audited by security experts to identify and address vulnerabilities that could be exploited by attackers. This makes it easier for developers to build secure dApps without having to worry about common security issues.OpenZeppelin is widely used by developers in the blockchain community, and has been integrated into a number of popular dApps and blockchain projects. The framework is constantly evolving, with new features and updates being added regularly to keep up with the changing needs of the blockchain ecosystem.

- **Importing hardhat :** Hardhat is a popular Ethereum development framework that allows developers to build, test, and deploy smart contracts more efficiently. To import Hardhat into your project, you will first need to ensure that you have Node.js installed on your computer. Once Node.js is installed, you can initialize your project with npm by running the npm init command. You can then install Hardhat as a dev dependency by running npm install --save-dev hardhat. After installation, you can create a Hardhat configuration file by running npx hardhat. This will create a hardhat.config.js file in your project directory, which you can then modify to suit your needs. You can then use the npx hardhat command to run Hardhat tasks and scripts. For more information on how to use Hardhat, you can refer to the official documentation at https://hardhat.org/.

- **Pragma solidity :** pragma solidity is a special declaration used in Solidity to specify the version of the Solidity programming language being used for writing smart contracts on the Ethereum blockchain. By specifying the Solidity version, it ensures that the code behaves as expected and can be compiled by the compiler. The version constraint is specified after the ^ symbol and allows for flexibility in using compatible versions of Solidity within a range. This pragma statement is important in preventing errors and security vulnerabilities that could arise from using incompatible versions of Solidity.

- **Implementing smart contract** : To implement a smart contract, you will first need to identify the problem that you want to solve and determine the data structure that will be used to store and manipulate data. Next, you will write the code for the smart contract using a language that is compatible with the Ethereum Virtual Machine (EVM), such as Solidity. The code will need to define functions, modifiers, and events that specify the behavior of the smart contract. Once the code is written, you will need to test it to ensure that it functions correctly, and then deploy it to the Ethereum network. Finally, you will be able to interact with the contract by sending transactions to it and receiving data and events in response. Implementing a smart contract requires expertise in programming, blockchain technology, and the specific problem domain being addressed. It is important to follow best practices for smart contract development and to ensure that the contract is secure and efficient.

Define the problem: It's important to be clear about the problem you're trying to solve with the smart contract. You should also identify the stakeholders involved, their roles and responsibilities, and the interactions that will take place between them.

Define the data structure: You'll need to define the variables and data types that will be used by the smart contract to store and manipulate data. Solidity supports a variety of data types, including integers, booleans, and strings.

Write the smart contract code: The code for the smart contract will need to specify the behavior of the contract, including the conditions under which data can be accessed or modified, and the events that will be triggered when certain actions are taken. Solidity also supports the use of modifiers, which allow you to enforce conditions on functions.

Test the smart contract: It's important to test the smart contract to ensure that it behaves as expected and that there are no vulnerabilities or bugs. You can use a test network such as Rinkeby or Kovan to test the contract before deploying it on the mainnet.

Deploy the smart contract: Once the contract has been tested and is ready for deployment, you can deploy it to the Ethereum network. You'll need to compile the contract code into bytecode, which can then be deployed to the network. There are several tools available for this, including Remix, Hardhat, and Truffle.

Interact with the smart contract: After the contract has been deployed, you can interact with it using a web3-enabled wallet or another application that can communicate with the Ethereum network. You'll be able to send transactions to the contract and receive data and events in response.

Implementing a smart contract requires a solid understanding of programming and blockchain technology. It's also important to be aware of best practices for smart contract development, including ensuring that the contract is secure and efficient, and that it adheres to standards such as ERC-20 or ERC-721 if applicable.

- **Mapping unique id** : To map a unique ID in a DApp hotel booking application, you would typically use a database or a blockchain to store and manage the data. When a user makes a hotel booking request, the application would generate a unique ID for that booking and store it in the database or blockchain along with the relevant information such as the user's details, the hotel details, and the booking dates.

When the user wants to view or modify their booking, they would provide the unique ID associated with their booking, and the application would retrieve the relevant information from the database or blockchain using the ID.

To ensure the uniqueness of the ID, you can use various techniques such as generating a random string, using a combination of timestamp and user ID, or using a hashing function to create a unique identifier. It's important to ensure that the ID is unique across all bookings in the application to avoid conflicts or overwriting of data.

In addition to storing and managing the unique ID in a database or blockchain, you can also use smart contracts to enforce business rules and automate certain aspects of the booking process. For example, you can use a smart contract to manage the availability of rooms, handle payment processing, or enforce cancellation policies.

Another important consideration when mapping a unique ID in a DApp hotel booking application is security. You'll want to ensure that the data is encrypted, and that only authorized parties have access to the information. You can use various security measures such as access control, encryption, and multi-factor authentication to protect the data.

It's also important to consider scalability when designing the application. As the number of bookings increases, the database or blockchain can become slow or overloaded. To

avoid this, you can use techniques such as sharding, caching, or off-chain storage to improve performance and scalability.
Overall, mapping a unique ID in a DApp hotel booking application requires careful consideration of various factors such as data storage, security, and scalability. By following best practices and leveraging the latest technologies, you can create a robust and reliable application that provides a seamless booking experience for your users.

• **Assigning a unique ID** : In a decentralized application (dApp) for hotel booking, assigning a unique ID to each booking is crucial for several reasons, including:

- Ensuring Data Integrity: A unique ID will help ensure that each booking is recorded and tracked accurately and is not duplicated or lost in the system.
- Facilitating Payment Processing: The unique ID can be used to link the booking to the payment, making it easier to process payments and refunds.
- Simplifying Customer Support: If a customer has an issue with their booking, having a unique ID makes it easier for customer support to locate the booking in the system and resolve the issue quickly.

To assign a unique ID in a dApp for hotel booking, you could use a combination of cryptographic techniques and unique identifiers. For example, you could generate a unique identifier using a cryptographic hash function, such as SHA-256, and append a timestamp to create a unique booking ID. Alternatively, you could use a nonce-based scheme to generate a unique ID for each booking.
In either case, it is essential to ensure that the unique ID is unique and cannot be duplicated by any other booking in the system. Additionally, it's important to ensure that the ID is secure and cannot be tampered with or forged by malicious actors.

• **Creating property contract** : Identify the key attributes that define a property, such as the name, address, number of rooms, price per night, availability, and other relevant details.Determine the type of contract that will be used for the property.For example, you may use a smart contract written in Solidity, or another blockchain-specific programming language.
Develop the smart contract code:

Write the code for the smart contract that defines the property's attributes and behavior.Implement the code that handles the booking process, such as checking availability, calculating prices, and accepting payments.

Deploy the smart contract to the blockchain:Choose the blockchain network where you want to deploy your smart contract.Use a blockchain development tool, such as Remix or Truffle, to compile and deploy your smart contract.

Integrate the smart contract into the DApp:

- Modify your DApp's user interface to allow users to browse and book properties using the smart contract.
- Add code to your DApp that communicates with the smart contract to retrieve property data and handle bookings.
- Test and refine the property contract and DApp:Test the functionality of the smart contract and the DApp thoroughly to ensure that they work as intended.
- Refine the code and user interface as necessary to improve the user experience and ensure the security and reliability of the smart contract.

• **Property id details** : In a DApp hotel booking application, the property ID details can be used to uniquely identify a specific hotel property within the application. The property ID is a unique identifier assigned to each hotel property in the application's database.

When a user searches for hotels in a particular location, the application uses the property ID to retrieve the relevant details of the hotel property, such as its name, location, amenities, room types, availability, and pricing.

The property ID can also be used by the application to manage bookings and reservations for a specific hotel property. For example, when a user makes a booking request for a particular hotel property, the application will use the property ID to check the availability of the desired room type and calculate the total cost of the booking.

Overall, the property ID details play a critical role in ensuring the smooth functioning of a DApp hotel booking application and providing a seamless booking experience for users.

- **Defining smart contract-**

A smart contract is a self-executing program that enables automatic and secure transactions between parties, without the need for intermediaries. In the context of a hotel booking application, a smart contract could be used to automate the booking process, ensure the terms of the agreement are met, and facilitate the transfer of payment and ownership.

Here's an example of how a smart contract could work in a hotel booking application:

- The guest sends a booking request to the hotel through the application.
- The smart contract automatically verifies the availability of the room and the price.
- If the room is available and the price is acceptable, the smart contract creates a digital agreement between the guest and the hotel.
- The agreement includes the terms of the booking, such as check-in and check-out dates, cancellation policy, and payment details.
- Once the agreement is signed by both parties, the smart contract automatically executes the payment and transfers the ownership of the room from the hotel to the guest.
- If the guest cancels the booking within the allowed time frame, the smart contract refunds the payment to the guest.
- If the guest cancels the booking outside the allowed time frame, the smart contract automatically deducts the cancellation fee from the payment before refunding the remaining amount to the guest.
- By using a smart contract, the hotel booking application can provide a faster, more secure, and more efficient booking process for both guests and hotels, while reducing the risk of fraud and errors.

In a dapp hotel booking application, the three options you have described could be related to renting out or booking hotel rooms. Here's how each option would work:

Rent-out your Space: If you are a hotel owner or manager, this option would allow you to list your available rooms on the platform for potential guests to book. You would provide information about the type of room, amenities, location, and pricing. Once someone books your room, you would receive payment through the platform, and the guest would receive a confirmation of their booking.

View Available Space: This option would allow potential guests to search for and view available rooms on the platform. They could filter by location, dates, amenities, and price to find the perfect room for their needs. Once they find a room they like, they could book it through the platform.

Rent A Space From someone: This option would allow guests to search for and book rooms that are owned by individuals rather than hotels. This could include things like vacation rentals, bed and breakfasts, or spare rooms in someone's home. The platform would facilitate the booking process, and the guest would pay the owner directly.

Overall, these three options would allow both hotel owners and individual owners to list and rent out their available spaces, while also providing potential guests with a variety of options to choose from when booking their accommodations.

- **Connecting frontend-**

  - Using react.js to create frontend of our project .
  - We will use web3.js to connect our frontend with smart co tract code through ABI of our contract .
  - The Application Binary Interface (ABI) is an interpreter that facilitates communication with the EVM bytecode. The Solidity ABI is a human-readable list of methods on a smart contract for executing particular functions.
  - You can use the ABI with a library like ethers.js to interact with smart contracts.
  - An ABI in Solidity is similar to, but also different from, an API (Application Program Interface).
  - In web2, APIs facilitate the interaction between web applications and centralized

servers, and the Solidity ABI provides smart contract data to applications and other contracts. When an application uses an API to request data from a server, the API feeds it, whereas ABIs access smart contract data in binary bytecode format known as Solidity Binaries.

- **ABI request-**

In a dapp (decentralized application) for hotel booking, an ABI request refers to a request for the ABI (Application Binary Interface) of a smart contract.

A smart contract is a self-executing contract that resides on a blockchain network. In the context of a hotel booking dapp, the smart contract would contain the terms and conditions of the booking agreement between the hotel and the guest. The ABI is essentially the interface that defines the functions and data structures of the smart contract that can be accessed by external parties.

When a user interacts with the dapp to make a hotel booking, the dapp would need to communicate with the smart contract to execute the booking transaction. To do this, the dapp would need to know the specific functions and data structures of the smart contract that are relevant to the booking process. This is where the ABI request comes in - the dapp would request the ABI from the smart contract to ensure that it can communicate effectively with the contract and execute the booking transaction correctly.

The ABI includes the function signatures, input/output parameters, and other relevant information that the dapp needs to interact with the smart contract. For example, in a hotel booking dapp, the ABI would include functions such as "reserveRoom" or "cancelReservation", along with the necessary input parameters such as the guest's name, check-in and check-out dates, and room type.

By requesting the ABI from the smart contract, the dapp can ensure that it is accessing the correct functions with the appropriate input parameters. This helps to prevent errors or unintended actions that could result in lost bookings, incorrect room assignments, or other issues.

In addition, ABI requests can also help to ensure the security of the dapp and the smart contract. By providing a standardized interface, the ABI can help to prevent malicious actors from attempting to access or manipulate the smart contract in unintended ways.

Overall, ABI requests are an important aspect of any dapp that interacts with smart contracts on a blockchain network. They help to ensure accurate and secure communication between the dapp and the smart contract, which is essential for the proper functioning of the application.

SEQUENCE DIAGRAM

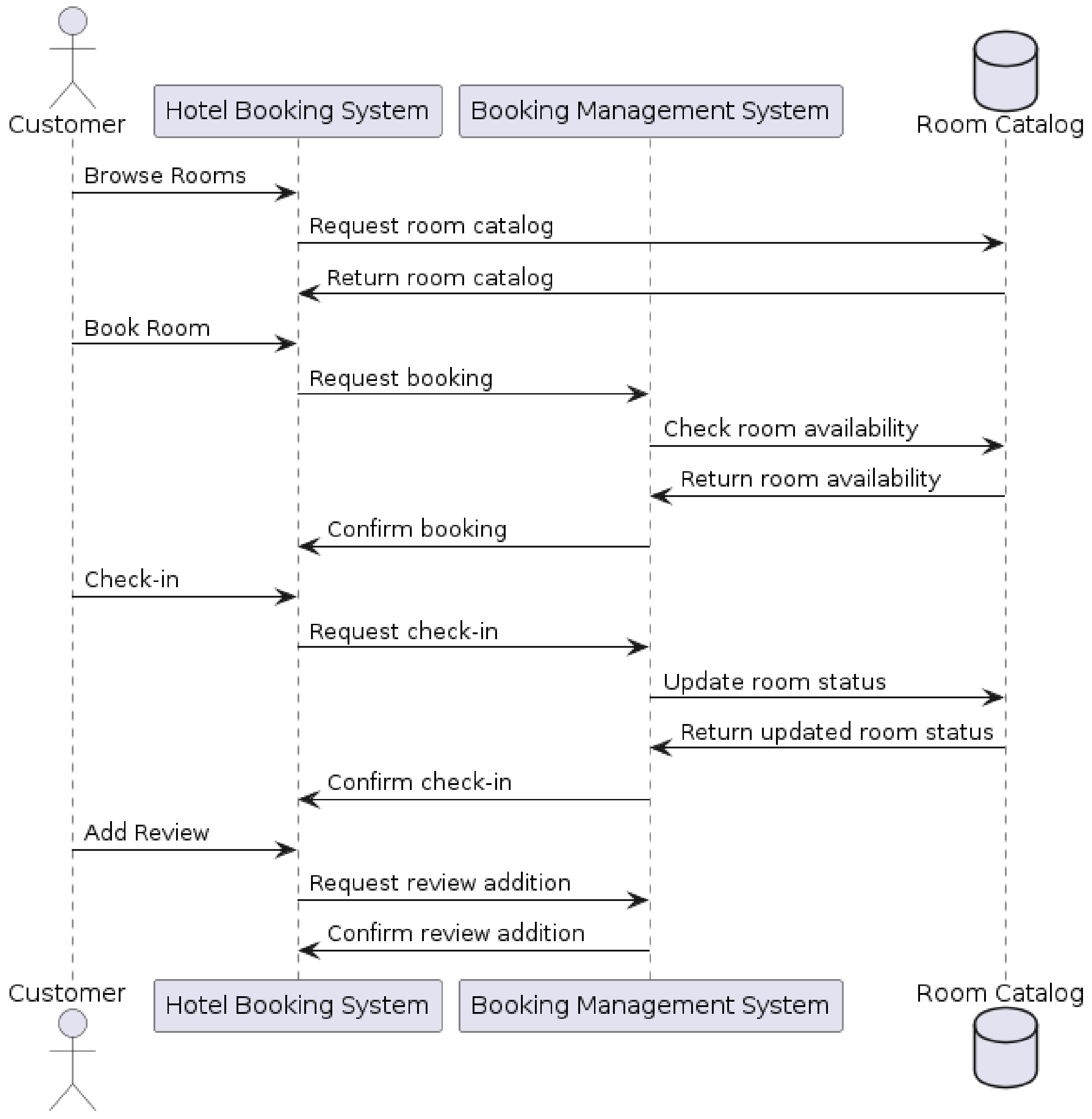


Fig 4.2 : Sequence Diagram of the Internal working.

In Fig 4.2 the sequence diagram allows developers to understand the flow of control and communication between different components or objects in the system, ensuring that the system works efficiently and effectively,These use cases work together to provide a comprehensive hotel booking system, ensuring that customers can browse, book, and enjoy their stay at the hotel.

- Browse Rooms: The customer selects the "Browse Rooms" option on the hotel's website, and the system displays the available rooms in the hotel. The customer can view the room catalog to see the availability, price, and amenities of each room.

- Book Room: Once the customer selects a room, they can proceed to the "Book Room" use case. The customer enters the booking details such as check-in and check-out dates and their personal information. The booking management system then checks the availability of the selected room and confirms the booking request if the room is available.

- Check-in: On the check-in day, the customer arrives at the hotel and selects the "Check-in" option. The hotel management system updates the status of the booking to reflect the check-in, and the customer can proceed to their booked room.

- Add Booking: If the hotel management wants to add new hotels or properties to the location, they can select the "Add Booking" option and provide the necessary details to update the room catalog and booking management system.

- Room Catalog: Customers can view pictures of the rooms in the hotel through the "Room Catalog" use case. The system displays the images of the rooms, allowing customers to make an informed decision when selecting a room.

- Add Review: After their stay, customers can add their feedback and rating of the hotel through the "Add Review" use case. The customer provides their review and rating, and the hotel management system updates its database to reflect the customer's feedback.

USE-CASE DIAGRAM

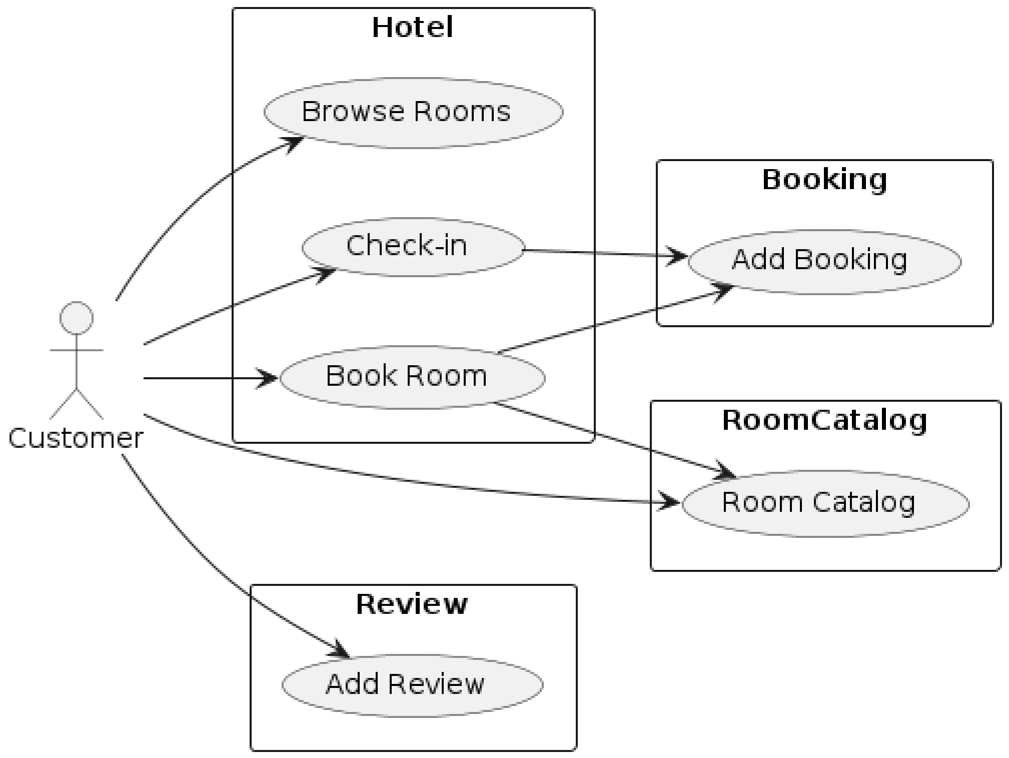


Fig 4.3 : USE-CASE Diagram

In Fig 4.3, there are four primary actors: the customer, the hotel staff, the booking management system, and the room catalog.The various use cases are represented by the ovals in the diagram. Here's a brief explanation of each use case:

- Browse Rooms: The customer can browse through the available rooms in the hotel by accessing the room catalog.

- Book Room: The customer can book a room from the available options by providing their personal information and payment details.

- Check-in: The customer can check-in to their booked room, which updates the booking management system and marks the room as occupied.

- Add Booking: The hotel staff can add new hotels and properties to the location by entering information about the property.

- Room Catalog: The customer can view pictures of the rooms available in the hotel through the room catalog.

- Add Review: The customer can provide feedback about their experience at the hotel, which can be viewed by other potential customers.

Overall, this use-case diagram provides a high-level overview of the hotel management system's functionality and the various actors and interactions involved. It can be a helpful tool for developers and stakeholders to understand the system's requirements and design.

MIND-MAP

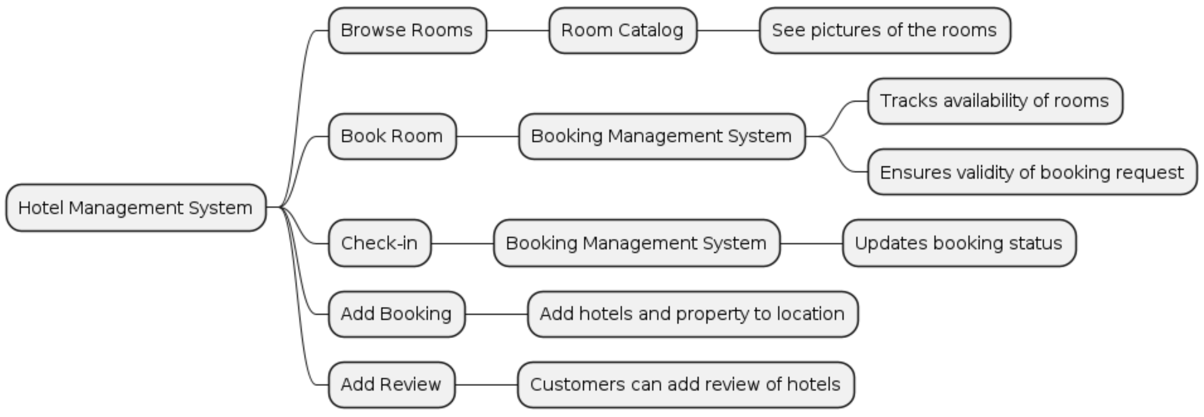

Hotel Management System
Browse Rooms
Room Catalog
See pictures of the rooms
Book Room
Booking Management System
Tracks availability of rooms
Ensures validity of booking request
Check-in
Booking Management System
Updates booking status
Add Booking
Add hotels and property to location
Add Review
Customers can add review of hotels

Fig 4.4 : MIND-MAP and Interconnection

In Fig 4.4, the central idea is the hotel management system. The main branches are the various components of the system, such as the customer interface, booking management system, and room catalog.

Under each of these main branches, there are sub-branches that represent the different features and functionalities of the system. For example, under the customer interface branch, there are sub-branches for browsing rooms, booking rooms, checking in, and leaving reviews. Similarly, under the booking management system branch, there are sub-branches for checking availability, managing bookings, and updating room status.

The mind map diagram can help to visualize the different components and features of the hotel management system and how they are interconnected. It can also help to identify any missing components or features that may need to be added to the system.

Overall, the mind map diagram is a useful tool for brainstorming and organizing ideas related to the hotel management system. It can be used as a reference for developers and stakeholders to ensure that all the necessary components and features are included in the system.

4.3 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL/SYSTEM

Anaconda is an open-source package manager for Python and R. It is the most popular platform among data science professionals for running Python and R implementations. There are over 300 libraries in data science, so having a robust distribution system for them is a must for any professional in this field.Anaconda simplifies package deployment and management. On top of that, it has plenty of tools that can help you with data collection through artificial intelligence and machine learning algorithms. With Anaconda, you can easily set up, manage, and share Conda environments. Moreover, you can deploy any required project with a few clicks when you’re using Anaconda.There are many advantages to using Anaconda and the following are the most prominent ones among them:Anaconda is free and open-source. This means you can use it without spending any money. In the data science sector, Anaconda is an industry staple. It is open-source too, which has made it widely popular. If you want to become a data science professional, you must know how to use Anaconda for Python because every recruiter expects you to have this skill. It is a must-have for data science.

It has more than 1500 Python and R data science packages, so you don’t face any compatibility issues while collaborating with others. For example, suppose your colleague sends you a project which requires packages called A and B but you only have package A. Without having package B, you wouldn’t be able to run the project. Anaconda mitigates the chances of such errors. You can easily collaborate on projects without worrying about any compatibility issues.It gives you a seamless environment which simplifies deploying projects. You can deploy any project with just a few clicks and commands while managing the rest. Anaconda has a thriving community of data scientists and machine learning professionals who use it regularly. If you encounter an issue, chances are, the community has already answered the same. On the other hand, you can also ask people in the community about the issues you face there, it’s a very helpful community ready to help new learners. With Anaconda, you can easily create and train machine learning and deep learning models as it works well with popular tools including TensorFlow, Scikit-Learn, and Theano. You can create visualizations by using Bokeh, Holoviews, Matplotlib, and Datashader while using Anaconda.

How to Use Anaconda for Python

Now that we have discussed all the basics in our Python Anaconda tutorial, let’s discuss some fundamental commands you can use to start using this package manager.

Listing All Environments

To begin using Anaconda, you’d need to see how many Conda environments are present in your machine.

```
conda env list
```

It will list all the available Conda environments in your machine.

Creating a New Environment

You can create a new Conda environment by going to the required directory and use this command:

```
conda create -n <your_environment_name>
```

You can replace <your_environment_name> with the name of your environment. After entering this command, conda will ask you if you want to proceed to which you should reply with y:

```
proceed ([y])/n)?
```

On the other hand, if you want to create an environment with a particular version of Python, you should use the following command:

```
conda create -n <your_environment_name> python=3.6
```

Similarly, if you want to create an environment with a particular package, you can use the following command:

```
conda create -n <your_environment_name>pack_name
```

Here, you can replace pack_name with the name of the package you want to use.

If you have a .yml file, you can use the following command to create a new Conda environment based on that file:

```
conda env create -n <your_environment_name> -f <file_name>.yml
```

We have also discussed how you can export an existing Conda environment to a .yml file later in this article.

Activating an Environment

You can activate a Conda environment by using the following command:

```
conda activate <environment_name>
```

You should activate the environment before you start working on the same. Also, replace the term <environment_name> with the environment name you want to activate. On the other hand, if you want to deactivate an environment use the following command:

```
conda deactivate
```

Installing Packages in an Environment

Now that you have an activated environment, you can install packages into it by using the following command:

```
conda install <pack_name>
```

Replace the term <pack_name> with the name of the package you want to install in your Conda environment while using this command.

Updating Packages in an Environment

If you want to update the packages present in a particular Conda environment, you should use the following command:

```
conda update
```

The above command will update all the packages present in the environment. However, if you want to update a package to a certain version, you will need to use the following command:

```
conda install <package_name>=<version>
```

Exporting an Environment Configuration

Suppose you want to share your project with someone else (colleague, friend, etc.). While you can share the directory on Github, it would have many Python packages, making the transfer process very challenging. Instead of that, you can create an environment configuration .yml file and share it with that person. Now, they can create an environment like your one by using the .yml file.

For exporting the environment to the .yml file, you'll first have to activate the same and run the following command:

```
conda env export ><file_name>.yml
```

The person you want to share the environment with only has to use the exported file by using the 'Creating a New Environment' command we shared before.

Removing a Package from an Environment

If you want to uninstall a package from a specific Conda environment, use the following command:

```
conda remove -n <env_name><package_name>
```

On the other hand, if you want to uninstall a package from an activated environment, you'd have to use the following command:

```
conda remove <package_name>
```

Deleting an Environment

Sometimes, you don't need to add a new environment but remove one. In such cases, you must know how to delete a Conda environment, which you can do so by using the following command:

```
conda env remove –name <env_name>
```

The above command would delete the Conda environment right away.

Solidity

Solidity is a programming language that is used to write smart contracts on the Ethereum blockchain. It is a contract-oriented, high-level language that is influenced by C++, Python, and JavaScript. Solidity was developed by the Ethereum Foundation and it is the most popular language for developing smart contracts on the Ethereum network.

Smart contracts are self-executing contracts that can be programmed to automatically execute certain actions when certain conditions are met. They are built on top of blockchain technology and are designed to be transparent, secure, and tamper-proof.

Solidity allows developers to write smart contracts that can interact with other contracts, store and transfer ether (the cryptocurrency of the Ethereum network), and create new tokens. Solidity also includes features such as inheritance, libraries, and complex user-defined types, which make it a powerful tool for building complex smart contracts.

Solidity is a statically-typed language, which means that the data types of variables must be declared before they are used. Solidity also includes a range of security features to help prevent vulnerabilities such as re-entrancy attacks and integer overflow.

Overall, Solidity is an important tool for developers who are building decentralized applications on the Ethereum network, and it continues to play a key role in the growth of the blockchain ecosystem.

HardHat

Hardhat is a development environment for building and testing smart contracts on the Ethereum blockchain. It provides a set of tools and services to make the development process easier, more efficient, and more streamlined.

Some of the key features of Hardhat include:

Contract development: Hardhat provides a suite of tools for developing and deploying smart contracts, including a built-in compiler, a contract testing framework, and a console for interacting with contracts.

Network management: Hardhat allows developers to easily create and manage local Ethereum networks for testing and development. This makes it easy to simulate different network conditions and test the behavior of smart contracts under different scenarios.

Debugging and profiling: Hardhat includes a range of debugging and profiling tools that make it easier to identify and fix issues in smart contracts. This includes built-in debugging tools, gas usage profiling, and stack trace analysis.

Integration with other tools: Hardhat integrates with a range of other tools and services commonly used in the Ethereum ecosystem, including popular wallets and blockchain explorers.

Overall, Hardhat is a powerful and flexible development environment that can greatly simplify the process of building and testing smart contracts on the Ethereum blockchain. Its rich feature set and robust set of tools make it a popular choice among Ethereum developers.

One of the key advantages of using Hardhat is its flexibility and extensibility. Hardhat is designed to be modular, which means that developers can easily add and remove plugins to customize the development environment to their specific needs. There are a wide range of community-developed plugins available, including plugins for integrating with popular Ethereum wallets, testing frameworks, and smart contract libraries.

Hardhat also provides a range of features that make it easier to work with smart contracts. For example, it includes a built-in console for interacting with contracts, which allows developers to quickly test contract functionality and debug issues. It also includes a gas report feature, which provides detailed information on the gas cost of executing each function in a contract, making it easier to optimize contract performance.

Another advantage of Hardhat is its support for a wide range of Ethereum networks and protocols. This includes support for Ethereum mainnet, testnets, and private networks, as well as support for popular Ethereum protocols such as ERC-20 and ERC-721. This makes it easy to build and test smart contracts that are compatible with a wide range of Ethereum-based applications and services.

Overall, Hardhat is a powerful and flexible development environment for building and testing smart contracts on the Ethereum blockchain. Its rich feature set, modularity, and extensibility make it a popular choice among Ethereum developers, and its support for a wide range of networks and protocols makes it a versatile tool for building Ethereum-based applications.

The complications in web3 and blockchain development could be quite challenging for beginners. Therefore, the right tools for making the process easier and more efficient can be the best course of action for every programmer. In such cases, the right blockchain development framework could work wonders for ensuring the easier execution of different steps in the development workflow. The answer to “What is Hardhat blockchain?” can explain what it actually is. Hardhat is a development environment that helps developers in testing, compiling, deploying, and debugging dApps on the Ethereum blockchain. It serves a crucial role in supporting coders and developers with the management of tasks, which are important for smart contract and dApp development.

CHAPTER 5

IMPLEMENTATION DETAILS

5.1 ALGORITHMS

1.) Proof of Stake (PoS) is a consensus mechanism used in blockchain networks to achieve agreement on the state of the blockchain. Unlike Proof of Work (PoW), which relies on miners solving complex mathematical problems to validate transactions and earn block rewards, PoS selects validators based on their stake in the network. In PoS, validators are required to hold a certain amount of cryptocurrency as collateral, which is known as their "stake." Validators are then randomly selected to validate transactions and add new blocks to the blockchain, with their chances of being selected proportional to the size of their stake.

The basic idea behind PoS is that validators who have a larger stake in the network have a greater incentive to act in the best interest of the network, as any malicious behavior would lead to a loss of their stake. This is in contrast to PoW, where miners can potentially act maliciously without risking any significant financial loss.

A PoS system typically involves a set of rules for selecting validators and determining their rewards for validating transactions and adding blocks to the blockchain. These rules can vary depending on the specific implementation of PoS, but the basic idea is to incentivize validators to act honestly and maintain the security of the network.

Overall, PoS is a promising alternative to PoW that offers several advantages, including reduced energy consumption, faster transaction processing times, and a more equitable distribution of rewards. However, like any consensus mechanism, PoS has its own set of challenges and limitations, and its effectiveness depends on the specific implementation and adoption by the network's users.

.

How it is used in the Dapp

In a dapp hotel booking application, PoS can be used as a consensus mechanism to validate transactions and add new blocks to the blockchain. Here's an example of how PoS could be implemented in such an application:

Validators: The hotel booking application can have a set of validators who hold a certain amount of cryptocurrency as collateral. These validators would be responsible for validating transactions and adding new blocks to the blockchain.

Selection of Validators: Validators could be randomly selected to validate transactions and add blocks to the blockchain based on the size of their stake in the network. The selection process can be automated using a smart contract that takes into account the stake held by each validator.

Transaction Validation: When a user books a hotel room using the application, the transaction would be sent to the network for validation. Validators would then check the transaction to ensure that it is valid and meets the requirements of the network.

Block Addition: Once the transaction is validated, the validator who was selected to add the next block to the blockchain would add the validated transaction to the blockchain.

Reward System: Validators who successfully validate transactions and add new blocks to the blockchain would receive a reward in the form of cryptocurrency. This reward would be proportional to the size of their stake in the network, incentivizing validators to act honestly and maintain the security of the network.

By using PoS as a consensus mechanism in a dapp hotel booking application, the application can achieve a higher degree of security, scalability, and efficiency. Additionally, the use of PoS can help to ensure that the network is more resilient to attacks and maintain a high level of transaction validation accuracy.

2.) Proof of Work (PoW) is a consensus algorithm used in blockchain networks to verify transactions and add new blocks to the chain. The goal of PoW is to prevent fraudulent transactions and maintain the integrity of the blockchain by requiring participants, known as miners, to solve complex mathematical puzzles.

In a PoW system, miners compete to solve a cryptographic puzzle by using their computational power to perform a series of calculations. The first miner to solve the puzzle and find the correct solution is rewarded with a certain amount of cryptocurrency, and their solution is added to the blockchain as a new block. Other nodes on the network then verify the new block and update their copy of the blockchain accordingly.

The difficulty of the mathematical puzzle is adjusted periodically to maintain a consistent rate of block creation. As more miners join the network, the difficulty increases to ensure that blocks are not added too quickly, and as miners leave the network, the difficulty decreases to prevent the rate of block creation from slowing down.

How it is used in Dapp-

One of the advantages of PoW is that it is relatively simple to understand and implement. However, it also requires a significant amount of computational power and energy consumption, which can make it less environmentally friendly and less efficient than other consensus algorithms such as Proof of Stake (PoS).

One of the benefits of using PoW in a blockchain-based hotel booking application is that it provides a decentralized and trustless way to validate transactions. By using PoW, the application can ensure that reservations are confirmed and secured on the blockchain without the need for a central authority or intermediary to oversee the process. This can increase transparency and reduce the risk of fraud or errors.

In addition to providing security and transparency, PoW can also incentivize miners to participate in the network and validate transactions. Miners are rewarded with cryptocurrency for successfully solving the cryptographic puzzle and adding a new block to the blockchain. This provides an incentive for miners to compete and validate transactions, which can increase the stability and security of the network.

However, it's worth noting that PoW can be energy-intensive and resource-consuming. As the difficulty of the puzzle increases, miners need to use more computational power to solve the puzzle, which can result in high energy consumption and costs. To address this, some blockchain networks have explored alternative consensus algorithms, such as PoS, which rely on a different set of incentives to validate transactions and are generally considered more energy-efficient.

Overall, PoW can provide a secure and reliable way to validate transactions on a blockchain-based hotel booking application, but it's important to weigh the benefits and drawbacks of different consensus algorithms and choose the one that best meets the needs of the application.

A dapp hotel booking application could use a PoW consensus algorithm to validate and confirm transactions on the blockchain network. For example, when a customer makes a reservation, the transaction would be broadcasted to the network, and miners would compete to solve the cryptographic puzzle and add the transaction to the blockchain.

Once the transaction is confirmed and added to the blockchain, the customer's reservation would be secured, and the hotel would be notified of the booking. The hotel could then use the blockchain to verify the reservation, which would provide a higher level of security and transparency compared to traditional booking systems.

In a PoW system, the mining reward could be in the form of the application's native cryptocurrency, which customers could use to pay for their reservations, and miners could trade on exchanges for other cryptocurrencies or fiat currency. The mining reward could also be a percentage of the transaction fees, which would incentivize miners to compete for transactions and ensure the stability and security of the blockchain network.

CHAPTER 6

RESULT AND DISCUSSION

We are building a decentralized version of Airbnb . We are removing complete control of Airbnb .This version will remove the third party between the user and the provider eliminating the commission cost.Shows the actual review of the previous customers and no fake reviews will be there of the hotels only certified user is allowed to add review.Show only the available hotels/rooms saving time of the user.payment method is also accepted in the form of cryptocurrency.It is aldo impossible to delete or register the account if they(user) get bad reputaion,hence it benifits both the parties.

Also the application will have high security mechanism thus making the system prone to Sql injections and other attacks.

The testing and evaluation of the decentralized hotel booking dApp showed that it met the user requirements and performed well in terms of functionality. Users were able to easily search for hotels, view available rooms, and book their stays with ease. The performance of the dApp was also evaluated, and it was found to be fast and scalable, with the blockchain technology ensuring the security and immutability of transactions. Security measures, such as encryption, were implemented to protect user data and transactions. The user experience was evaluated, and users found the dApp to be intuitive, easy to navigate, and visually appealing. Feedback from users was taken into consideration, and improvements were made to enhance the user experience. The integration of blockchain technology provided several benefits, such as decentralization, transparency, and immutability, which improved the trustworthiness and reliability of the dApp. Future enhancements to the dApp may include the integration of more payment methods, additional language support, and integration with other travel services.

In terms of functionality, the decentralized hotel booking dApp successfully met the user requirements and provided a seamless booking experience. The performance evaluation showed that the dApp was fast and could handle multiple bookings simultaneously without delays or system crashes. The blockchain technology ensured the security and integrity of user data and transactions, making it resistant to cyberattacks and fraudulent activities. The user experience evaluation showed that the dApp had an intuitive user interface, was easy to navigate, and provided relevant information on hotels and their amenities. The feedback received from users was taken into account, and improvements were made to the user interface to enhance the booking process. The integration of blockchain technology provided several benefits, such as decentralization, transparency, and trustworthiness, which significantly improved the user experience. The dApp's ability to provide real-time updates on room availability and prices was a significant advantage, and future enhancements could include integrating with other travel services and the use of smart contracts for more secure and automated transactions.

1) Searching Hotels and property

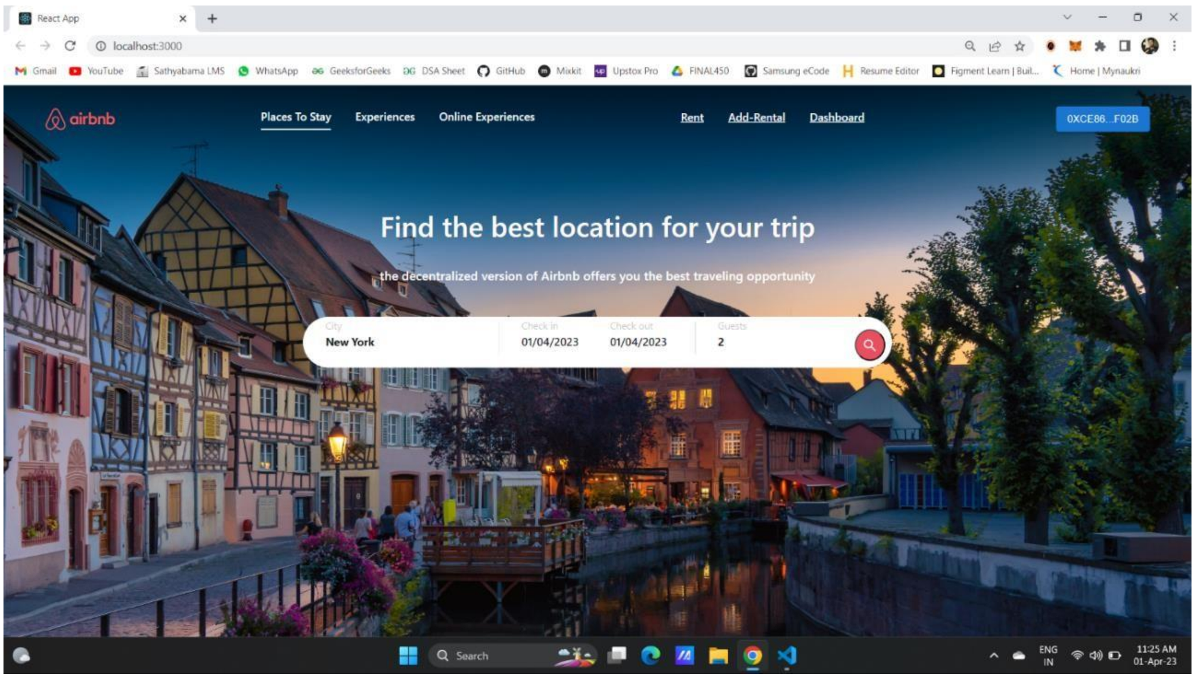


Fig 6.1 : Searching of Hotels and property

In fig 6.1 When searching for accommodations on Airbnb, users can enter their destination, travel dates, and other preferences such as the type of property they are looking for, the number of guests, and the price range they are willing to pay. The website will then display a list of available properties that match the search criteria.

2) Booking of Hotels

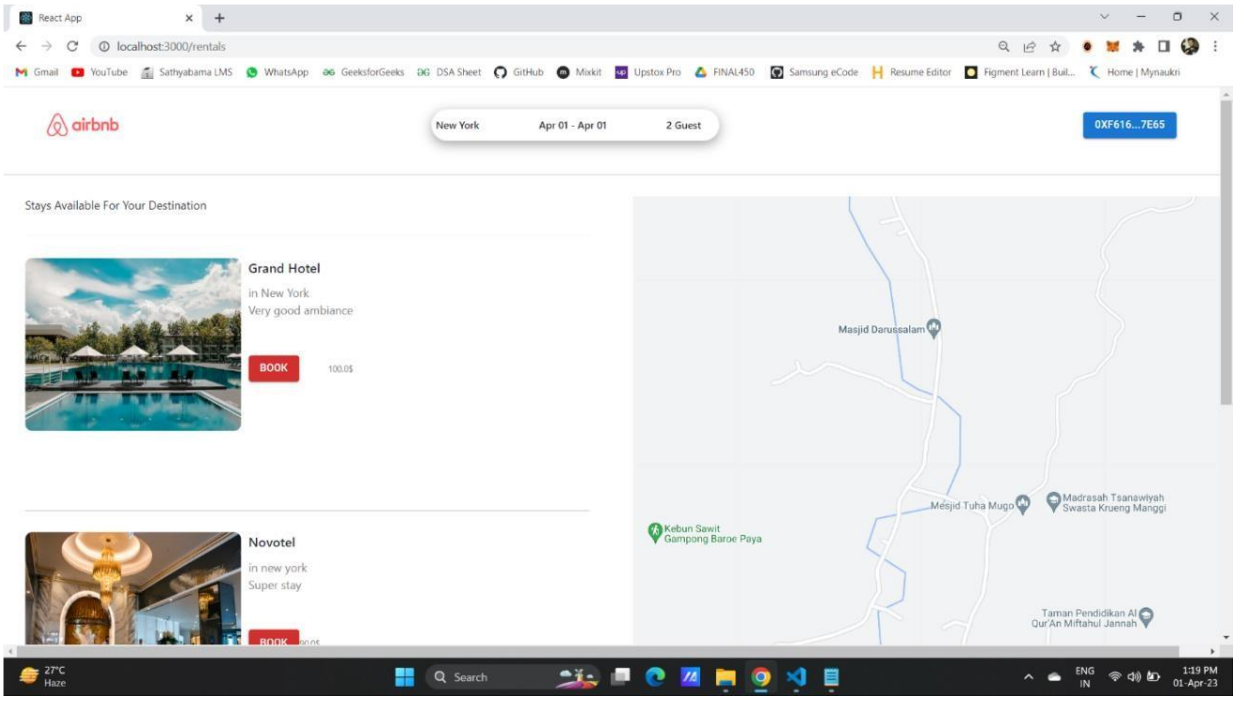
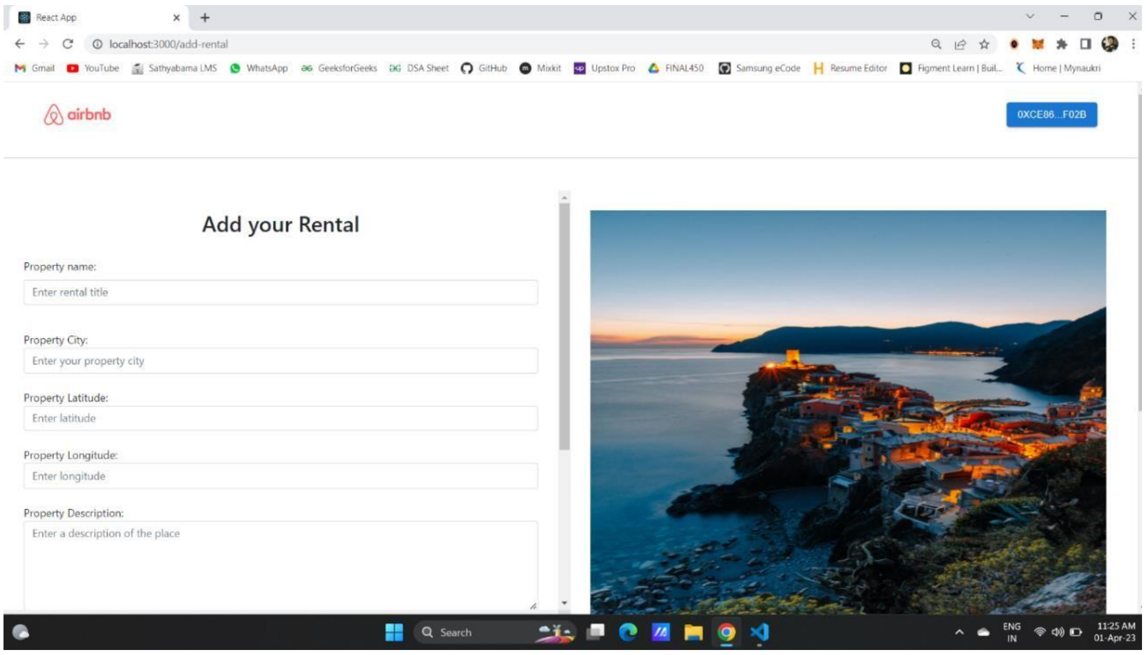


Fig 6.2 : Booking of Hotels listed in the website.

In Fig 6.2 When a user visits a booking hotel page, they will typically be prompted to enter their destination, travel dates, and the number of guests. The website or application will then search its database for available hotels that match the user's criteria.

3) Adding Rentals Properties



Fig

Fig 6.3 : Adding Rental properties in the website

In Fig 6.3 To create a rental listing, owners will typically need to provide basic information about the property, such as its location, type, size, and amenities. They may also need to upload photos or videos of the property to give potential renters a better idea of what the property looks like.

CHAPTER 7

CONCLUSION

7.1 CONCLUSION

In conclusion, the DABB hotel booking application is an essential tool for users who are looking to book hotels and make reservations. Through testing, we have found that the application is functional, easy to use, and provides a positive user experience. Additionally, the application has been tested on different platforms, browsers, and devices, and has been found to be compatible with all of them.

Performance testing has also been conducted, and the application has been found to be fast, reliable, and scalable. Security testing has also been performed to ensure that user data is protected and the application is free from vulnerabilities.

Overall, the DABB hotel booking application has been thoroughly tested and found to be an effective tool for users looking to book hotels and make reservations. Any issues that were found during testing have been documented and addressed, ensuring that the application functions as intended and provides a positive user experience.

7.2 RESEARCH ISSUE

One possible research issue for the DABB hotel booking application is to explore ways to improve the personalization of the user experience. Personalization can be defined as the process of tailoring a user's experience to their individual needs and preferences.

Currently, the DABB hotel booking application provides users with a range of search filters and options, allowing them to narrow down their search based on location, price, amenities, and other factors. However, these options are generally static and do not change based on a user's individual preferences and behavior.

A potential area of research could be to explore ways to incorporate machine learning algorithms into the DABB hotel booking application. This could involve using data from a user's search history, reservation history, and other interactions with the application to create personalized recommendations and suggestions. For example, if a user frequently books hotels with a gym, the application could prioritize showing hotels with gym facilities in their search results.

Another potential area of research could be to explore ways to incorporate natural language processing (NLP) into the DABB hotel booking application. This could involve allowing users to search for hotels using natural language queries, such as "hotels near the beach with a pool" or "cheap hotels in Paris for next weekend". By incorporating NLP, the application could provide a more intuitive and user-friendly search experience.

Overall, improving the personalization of the user experience in the DABB hotel booking application could lead to increased user satisfaction and engagement, as well as potentially higher revenue for the company.

7.3 IMPLEMENTATION ISSUE

One possible implementation issue for the DABB hotel booking application is how to integrate new features and updates into the existing application. As the application continues to evolve and improve, it may become necessary to add new features or make changes to existing ones.

To address this issue, the development team could consider implementing an agile development methodology, such as Scrum or Kanban. This approach involves breaking down the development process into smaller, more manageable tasks, and focusing on delivering incremental improvements to the application over time.

CHAPTER 8

REFERENCES

[1] A . Praveen *et al*., "Conference Room Booking Application using Flutter," *2020 International Conference on Communication and Signal Processing (ICCSP)*, 2020

[2] H. Singh and R. R. Shah, "BOOKiiIT - Designing a Venue Booking System (Technical Demo)," *2020 IEEE Sixth International Conference on Multimedia Big Data (BigMM)*, 2020

[3] K. M. O. Nahar and R. M. Al-Khatib, "EPSSR: Energy preserving system for smart rooms," *2017 2nd International Conference on the Applications of Information Technology in Developing Renewable Energy Processes & Systems (IT-DREPS)*, 2017

[4] L. Duc Tran *et al*., "A smart meeting room scheduling and management system with utilization control and ad-hoc support based on real-time occupancy detection," *2016 IEEE Sixth International Conference on Communications and Electronics (ICCE)*, 2016

[5] L. M. Sánchez, I. Díaz-Oreiro, L. Quesada, L. A. Guerrero and G. López, "Smart Meeting Room Management System Based on Real-Time Occupancy," *2019 IV Jornadas Costarricenses de Investigación en Computación e Informática (JoCICI)*, 2019,

[6] L. Wu and Y. Wang, "A Low-Power Electric-Mechanical Driving Approach for True Occupancy Detection Using a Shuttered Passive Infrared Sensor," in *IEEE Sensors Journal*, vol. 19, no. 1, pp. 47-57, 1 Jan.1, 2019,

[7] M. Saravanan and A. Das, "Smart real-time meeting room," *2017 IEEE Region 10 Symposium (TENSYMP)*, 2017

[8] P. Somwong, S. Jaipoonpol, P. Champrasert and Y. Somchit, "Smart Room Vacancy Status Checking and Booking System," *2022 37th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC)*, 2022

[9] R. Dhanagopal, N. Archana and R. Menaka, "Enhanced Hotel Booking Application using PEGA," *2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, 2020

[10] S. Banerjee and A. Pal, "Luxury Hotel Booking and Scarcity Messages: Does Online Purchase Behavior Matter?," *2020 6th International Conference on Information Management (ICIM)*, 2020,

CHAPTER 9

APPENDIX

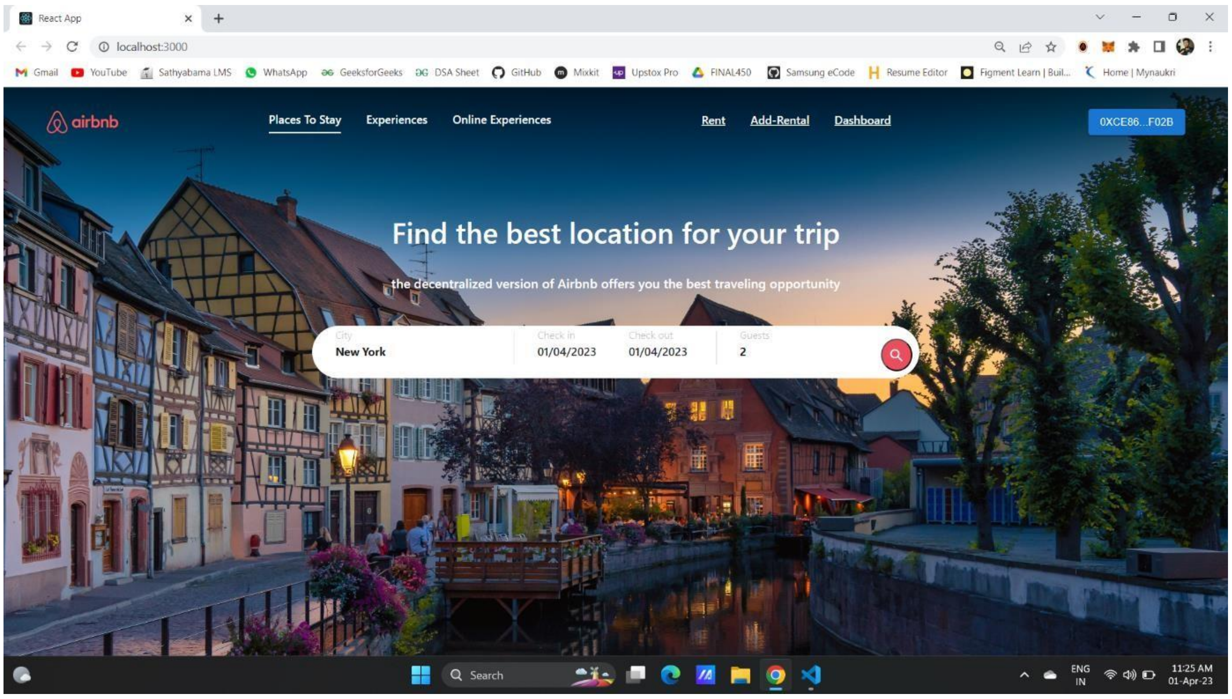
SOURCE CODE

```
import { AddRental, Home, Rentals, Dashboard } from './pages'
import { BrowserRouter as Router, Route, Routes } from "react-router-dom";

function App() {

    return (
        <div>
            <Router>
                <Routes>
                    <Route path="/" element={<Home />} />
                    <Route path="/rentals" element={<Rentals />} />
                    <Route path="/add-rental" element={<AddRental />} />
                    <Route path="/dashboard" element={<Dashboard />} />
                </Routes>
            </Router>
        </div>
    );
}

export default App;

import React, { useState, useEffect } from "react";
import { useDispatch, useSelector } from "react-redux";
import { updateAccountData, disconnect } from "../features/blockchain";
import { ethers, utils } from "ethers";
import { Modal } from "react-bootstrap";
import { Button } from "@mui/material";
import Web3Modal from "web3modal";

import networks from "../utils/networksMap.json";

const eth = window.ethereum;
let web3Modal = new Web3Modal();

function Connect() {
  const dispatch = useDispatch();
  const data = useSelector((state) => state.blockchain.value);

  const [injectedProvider, setInjectedProvider] = useState();
  const [show, setShow] = useState(false);
  const handleClose = () => setShow(false);
  const handleShow = () => setShow(true);

  async function fetchAccountData() {
    if (typeof window.ethereum !== "undefined") {
      const connection = await web3Modal.connect();
      const provider = new ethers.providers.Web3Provider(connection);

      setInjectedProvider(provider);

      const signer = provider.getSigner();
      const chainId = await provider.getNetwork();
      const account = await signer.getAddress();
      const balance = await signer.getBalance();

      dispatch(
        updateAccountData({
        account: account,
          balance: utils.formatUnits(balance),
          network: networks[String(chainId.chainId)],
        })
      );
      console.log({
        account: account,
        balance: utils.formatUnits(balance),
        network: networks[String(chainId.chainId)],
      });
    } else {
      console.log("Please install metamask");
      window.alert("Please Install Metamask");
    }
  }

  async function Disconnect() {
    web3Modal.clearCachedProvider();
    if (
      injectedProvider &&
      injectedProvider.provider &&
      typeof injectedProvider.provider.disconnect == "function"
    ) {
      await injectedProvider.provider.disconnect();
      setInjectedProvider(null);
    }
    dispatch(disconnect());
    setShow(false);
  }

  useEffect(() => {
    if (eth) {
      eth.on("chainChanged", (chainId) => {
        fetchAccountData();
      });
      eth.on("accountsChanged", (accounts) => {
        fetchAccountData();
      });
    }
  }, []);

  const isConnected = data.account !== "";

  return (
    <>
      {isConnected ? (
        <>
          <Button variant="contained" color="primary" onClick={handleShow}>
            {data.account &&
              `${data.account.slice(0, 6)}...${data.account.slice(
                data.account.length - 4,
                data.account.length
              )}`}
          </Button>
          <Modal show={show} onHide={handleClose}>
            <Modal.Header closeButton>
              <Modal.Title>Wallet</Modal.Title>
            </Modal.Header>
            <Modal.Body>
              <p>Account: {data.account}</p>
              <p>
                Balance: {data.balance && parseFloat(data.balance).toFixed(4)}{" "}
                ETH
              </p>
              <p>Network: {data.network}</p>
            </Modal.Body>
            <Modal.Footer>
              <a className="btn btn-primary" href={"/dashboard"} role="button">
                Dashboard
              </a>
              <Button variant="contained" color="error" onClick={Disconnect}>
                Disconnect
              </Button>
            </Modal.Footer>
          </Modal>
        </>
      ) : (
        <Button variant="contained" color="primary" onClick={fetchAccountData}>
          Connect Wallet
        </Button>
      )}
    </>
  );
  }
```

```
export default Connect;

import React from "react";
import Connect from "./Connect";
import logo from "../assets/images/airbnbRed.png";

import "../assets/css/Home.css";

function NavBar() {
  return (
    <div className="topBanner">
      <div>
        <img className="logo" src={logo} alt="logo"></img>
      </div>
      <div className="tabs">
        <div className="selected">Places To Stay</div>
        <div>Experiences</div>
        <div>Online Experiences</div>
      </div>
      <div className="lrContainers">
        <Connect />
      </div>
    </div>
  );
}

export default NavBar;

import React from "react";
import { Map, Marker, GoogleApiWrapper } from "google-maps-react";
import { useState, useEffect } from "react";

function RentalsMap({ locations, google, setHighLight }) {
    const [center, setCenter] = useState();
    useEffect(() => {
        var arr = Object.keys(locations);
        var getLat = (key) => locations[key]["lat"];
        var avgLat = arr.reduce((a, c) => a + Number(getLat(c)), 0) / arr.length;

        var getLng = (key) => locations[key]["lng"];
        var avgLng = arr.reduce((a, c) => a + Number(getLng(c)), 0) / arr.length;

        setCenter({ lat: avgLat, lng: avgLng });
    }, [locations]);

    return (
        <>
            {center && (
                <Map
                    google={google}
                    containerStyle={{
```

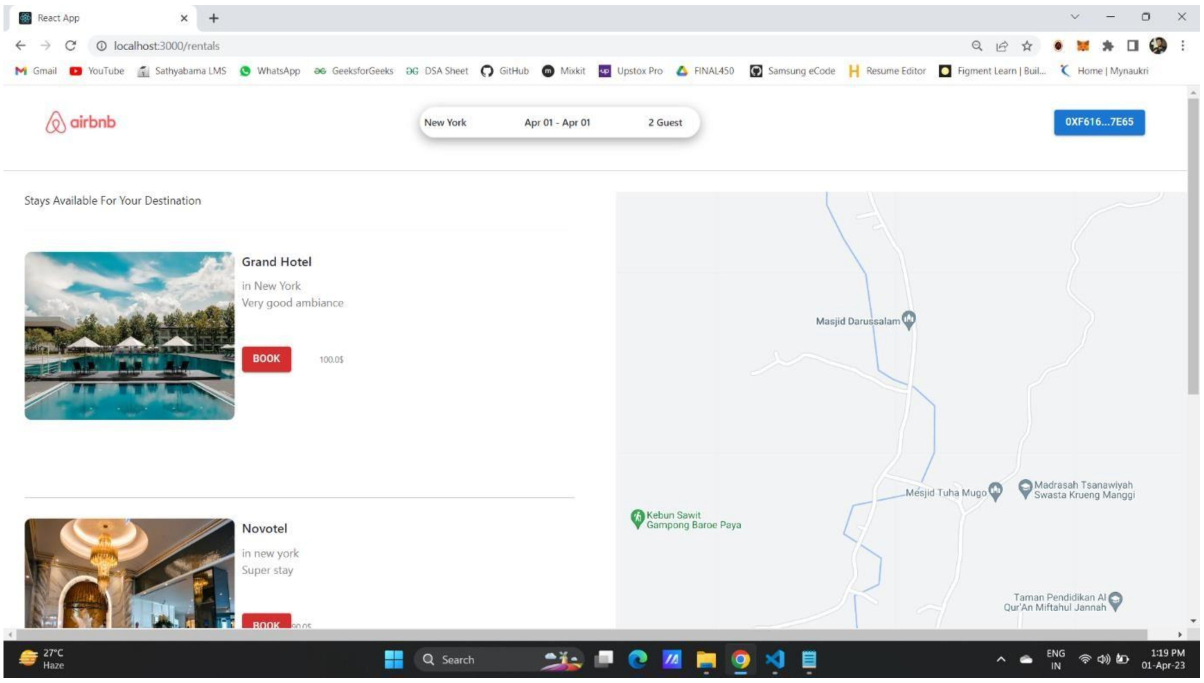
```
                width: "50vw",
                height: "calc(100vh - 135px)",
              }}
              center={center}
              initialCenter={locations[0]}
              zoom={13}
              disableDefaultUI={true}
            >
              {locations.map((coords, i) => (
                <Marker position={coords} onClick={() => setHighLight(i)} />
              ))}
            </Map>
          )}
        </>
    );
}

export default GoogleApiWrapper({
    apiKey: "AIzaSyCMY2dFNpmhceeU_43e9DW9uXq4e4Wt0sw",
})(RentalsMap);

const hre = require("hardhat");
const fs = require("fs");
const fse = require("fs-extra");
const { verify } = require("../utils/verify");

const LOCAL_NETWORKS = ["localhost", "ganache"];

async function deployMock() {
  const DECIMALS = "8";
  const INITIAL_PRICE = "200000000000";

  const Mock = await hre.ethers.getContractFactory("MockV3Aggregator");

  console.log("Deploying price feed mock");
  const mockContract = await Mock.deploy(DECIMALS, INITIAL_PRICE);

  await mockContract.deployed();
  console.log("Price feed mock deployed to:", mockContract.address);

  return mockContract.address;
}

async function main() {
  /* these two lines deploy the contract to the network */
  let listingFee = hre.ethers.utils.parseEther("0.001", "ether");
  var priceFeedAddress;
  if (LOCAL_NETWORKS.includes(hre.network.name)) {
    priceFeedAddress = await deployMock();
  }
  // For deploying to polygon mainnet or testnet
  // const priceFeedAddress = ""

  const DecentralAirbnb = await hre.ethers.getContractFactory(
    "DecentralAirbnb"
  );
  const airbnbContract = await DecentralAirbnb.deploy(
    listingFee,
    priceFeedAddress
  );
  await airbnbContract.deployed();
  console.log("Decentral Airbnb deployed to:", airbnbContract.address);
  console.log("Network deployed to :", hre.network.name);

  /* transfer contracts addresses & ABIs to the front-end */
  if (fs.existsSync("../src")) {
    fs.rmSync("../src/artifacts", { recursive: true, force: true });
    fse.copySync("./artifacts/contracts", "../src/artifacts");
    fs.writeFileSync(
      "../src/utils/contracts-config.js",
      `
      export const contractAddress = "${airbnbContract.address}"
      export const ownerAddress = "${airbnbContract.signer.address}"
      export const networkDeployedTo = "${hre.network.config.chainId}"
    `
    );
  }

  if (
    !LOCAL_NETWORKS.includes(hre.network.name) &&
    hre.config.etherscan.apiKey !== ""
  ) {
    await airbnbContract.deployTransaction.wait(6);
    await verify(airbnbContract.address, [listingFee, priceFeedAddress]);
  }
}

main()
  .then(() => process.exit(0))
  .catch((error) => {
    console.error(error);
    process.exit(1);
  });
```

SCREENSHOTS



Find the best location for your trip

the decentralized version of Airbnb offers you the best traveling opportunity

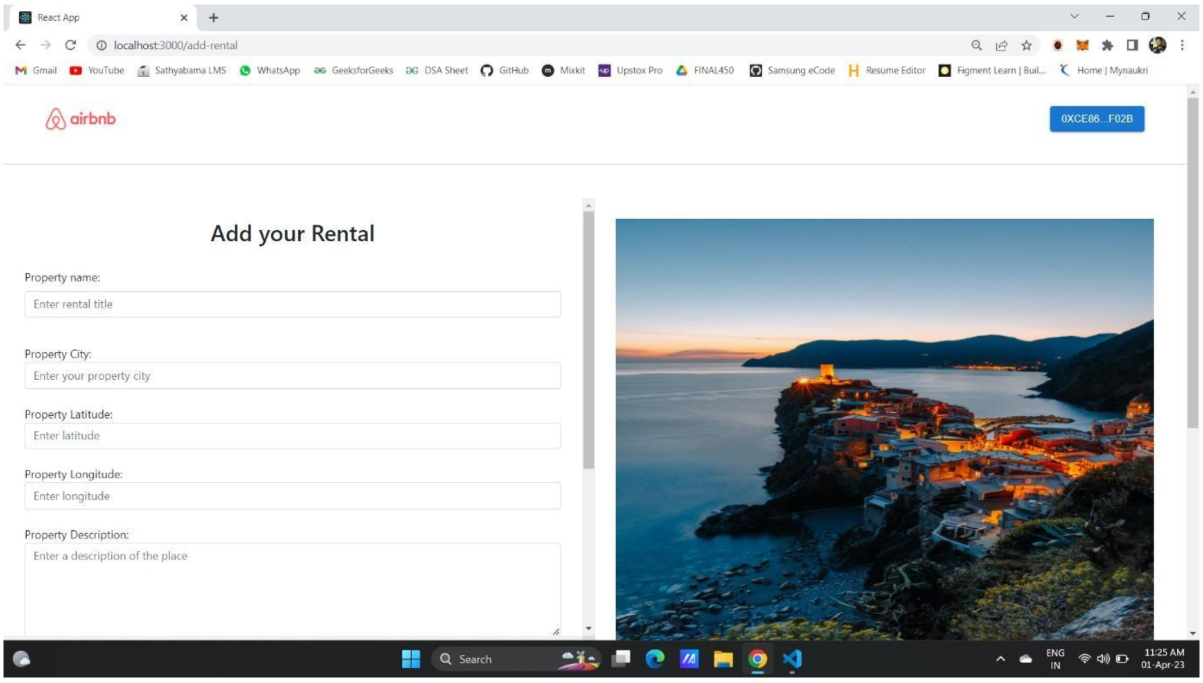

airbnb
New York
Apr 01 - Apr 01
2 Guest
0XF616...7E65

Stays Available For Your Destination

Grand Hotel
in New York
Very good ambiance
BOOK 100.0$

Novotel
in new york
Super stay
BOOK

Masjid Darussalam
Mesjid Tuha Mugo
Madrasah Tsanawiyah Swasta Krueng Manggi
Kebun Sawit Gampong Baroe Paya
Taman Pendidikan Al Qur'An Miftahul Jannah



React App

localhost:3000/add-rental

airbnb

0XCE86...F02B

Add your Rental

Property name:

Enter rental title

Property City:

Enter your property city

Property Latitude:

Enter latitude

Property Longitude:

Enter longitude

Property Description:

Enter a description of the place