Decentralized Application for Peer to Peer Buying and Selling of Land using GIS and Blockchain

Final Year Project UG 2019

By

Leader-321881 Mohammad Talal Farukh

Member 1-294449 – Fatima Zahra

Member 2-293546 – Asma Waheed

Member 3-288119 – Muhammad Usman Bakht

Institute of Geographical Information Systems

School of Civil and Environmental Engineering

National University of Sciences and Technology

Islamabad, Pakistan.

YEAR 2023

This is to certify that the

Final Year Project Titled

Decentralized Application for Peer to Peer Buying and Selling of Land Using GIS and Blockchain

Submitted by

Leader-321881 Mohammad Talal Farukh

Member 1-294449 – Fatima Zahra

Member 2-293546 – Asma Waheed

Member 3-288119 – Muhammad Usman Bakht

has been accepted towards the requirements.

for the undergraduate degree

in

BE Geoinformatics

_____

Dr. Ali Tahir

Associate Professor

School/Institute of SCEE, IGIS

National University of Sciences and Technology

Islamabad, Pakistan.

FORM UGP-OBE-1

INSTITUTE OF GEOGRAPHICAL INFORMATION SYSTEMS (IGIS)
UNDERGRADUATE FINAL PROJECT
(Formulation of Project Group and Advisor)

Date: 01/06/2023

Project Title: Decentralized Application for Peer to Peer Buying and Selling of Land using GIS and Blockchain

Project Advisor:
Name: Dr. Ali Tahir
Dept: Institute of Geographical Information System, SCEE

Project Members

1. Name (Group Leader): Muhammad Talal Farrukh CGPA: 3.54
   NUST Regn No: 321881 Signature: _____
2. Name: Muhammad Usman Bakht CGPA: 2.96
   NUST Regn No: 288119 Signature: _____
3. Name: Fatima Zahra CGPA: 3.00
   NUST Regn No: 294449 Signature: _____
4. Name: Asma Waheed CGPA: 3.28
   NUST Regn No: 293546 Signature: _____

Note: Group cannot be more than 4 students

_____ _____
Signature of Advisor Signature of Head of Department

APPROVAL

_____
Signature of Associate Dean

ACKNOWLEDGMENTS

We would like to extend our sincere appreciation and gratitude to all those who have contributed to the completion of this thesis.

We are indebted to our project supervisor, Dr. Ali Tahir, for his guidance throughout this research endeavor. We are grateful for his mentorship and dedication.

We would also like to express our heartfelt gratitude to our parents. Their constant support, both emotionally and financially, has been a driving force behind our academic pursuits. We are forever grateful for their sacrifices and the values they instilled in us.

Furthermore, we extend our gratitude to our professors and the faculty members whose profound knowledge and enthusiasm have enriched our learning experience.

Additionally, we would like to acknowledge the contributions of the participants whose cooperation and involvement were vital in gathering the necessary data and insights for the research.

We would be thankful to all our class fellows who supported us emotionally, and were there to guide us in every step.

This thesis would not have been possible without the collective efforts and support of these remarkable individuals. We are sincerely grateful to everyone for his/her contributions.

DEDICATION

This project is dedicated to the members of the team for their continuous hard work on a concept that is new to the world, throughout the year. We also dedicate this project to our peers and family members who stood by us throughout the journey, believed in us and continuously supported us.

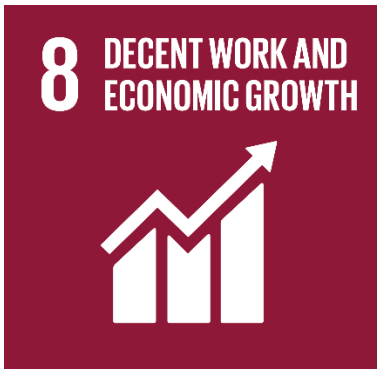
Thanks, a lot.

Table of Contents

List of Figures

SUSTAINABLE DEVELOPMENT GOALS CONTRIBUTION

United Nations established the Sustainable Development Goals (SDGs) in 2015 to address the world's most pressing social economic and environmental challenges and provide a framework for addressing these challenges. Our project meets 3 out of 17 SDGs.



ABSTRACT

This project endeavors to develop a comprehensive solution to rectify the inherent deficiencies present in Pakistan's land management system, which include issues such as inaccurate land records, high reliance on intermediaries, corruption, delays and inefficiencies, and high costs. This project is designed to enable direct peer-to-peer buying and selling of land in the real estate market by leveraging the power of blockchain technology and digital tokens. Through the application, we aim to eliminate intermediaries, which will ultimately reduce transaction costs for buyers and sellers. Additionally, the platform streamlines the process, reducing the amount of time and effort required. It guarantees secure and transparent transactions, making the entire process efficient, trustworthy, and seamless. Furthermore, by using Geographic Information System (GIS) technology, our application provides potential buyers with a comprehensive overview of the properties they are interested in, simplifying their decision-making process, and streamlining the buying and selling experience. By automating the land registration and ownership process using smart contracts, we aim to minimize the risk of fraud and disputes, providing a more reliable, secure, and efficient alternative. Our project covers the following SDGs, SDG 8 - Decent work and economic growth, SDG 9 - Industry, Innovation and Infrastructure and SDG 16 - Peace, Justice and Strong Institutions. Overall, our Decentralized Application (DApp) represents a promising solution for the real estate market in Pakistan, offering a revolutionary approach to buying and selling land that is more transparent, secure, and cost-effective.

INTRODUCTION

1.1. Background

The real estate market is quite a colossal industry representing a significant sector in the global as well as the local economy. The estimated market size is USD 3.69 trillion in 2021, encompassing many sectors such as residential, commercial, industrial as well as agricultural land. The market's growth is powered by various factors like urbanization, population growth, low-interest rates, and the rising demand for sustainable and intelligent buildings, which further enhance the industry's potential.

In the case of Pakistan, the real estate market plays a crucial role in contributing to the country's economy, and its size has been growing steadily, despite encountering several difficulties over the past decade. The Pakistan Bureau of Statistics reported that the real estate sector's contribution to the country's Gross Domestic Product (GDP) was around 13.5% in 2020, as opposed to 12.6% in 2019. Furthermore, the State Bank of Pakistan (SBP) estimates the real estate sector's contribution to the country's GDP to be approximately 2%, with an estimated value of PKR 5.2 trillion (USD 32 billion) in 2021 (Ahsan, 2017). Real Estate Activities (OD) has a significant 9.6% share in services and 5.6% share in GDP. We see how the real estate market in Pakistan has a considerable impact on the country's economy, with a total value of around PKR 13.1 trillion in 2020.

1.2. Rationale

In Pakistan, the real estate market, like many other countries, plays a significant role in contributing to the country's economy. Despite this, still the industry hit many downs and faced numerous challenges encompassing lack of transparency, insufficient regulations, shortage of affordable housing and a slowed demand in due to economic uncertainty, in the recent years. These challenges have led to a significant investment decline, decrease in property's value as well as economic activity, which ultimately affects the growth and development of the country's economy.

Moreover, these prevailing issues are not unique to Pakistan only, rather a global concern. The challenges faced by this sector in Pakistan are like those faced globally, including continuous inflation, a rising

interest rates, housing affordability, supply and demand dynamics, geopolitical risks, and changing consumer behavior. As a result, there is a pressing need to address these challenges and develop innovative solutions to revitalize the real estate market both nationally and globally, ultimately contributing to sustainable economic growth and development.

According to a report by Grand View Research, the global real estate market is expected to grow at a compound annual growth rate (CAGR) of 3.4% from 2021-2028.

The current process of buying and selling real estate can be complex and time-consuming, involving intermediaries such as real estate agents, title companies, banks, and lawyers. This can result in several challenges that affect both buyers and sellers. Some of the main challenges include:

- Lack of Transparency: This being one of the biggest issues in market leads to numerous other complicated challenges such as fraudulent activities, inaccurate property description and financial & legal risks for buyers (Ahsan, 2017). Limited access to information, such as property transactions, prices, and ownership, can make it difficult for buyers and sellers to make informed decisions, resulting in a lack of trust and a slow growth rate for the market. Therefore, it is crucial to address these challenges and promote greater transparency by implementing measures such as creating central registries of property ownership and transaction records, improving access to property information for the public, and strengthening regulations around property transactions. By doing so, the real estate market can become more efficient, competitive, and secure for all stakeholders.

- Corruption and bribery: This contribute to a lack of transparency in the real estate market. For example, officials may demand bribes in exchange for expediting authorizations and approvals or making favorable zoning decisions.

- High Transaction Costs: The process of buying and selling as we see often involves many complexities, making it slow and cumbersome. There is a continuous involvement of multiple intermediaries in all steps of the process such as real estate agents, title companies, lawyers etc., leading to increased bureaucracy and paperwork which further increases the costs and time. We must not forget the barrier that is created for sellers and specifically for buyers with the associated costs and continuous increase in them due to the above-mentioned causes. Promoting greater efficiency and transparency in the transaction process, by using technology and digital platforms for streamlined

process provision is necessary for addressing these challenges. By doing so, it can become easier and more affordable for buyers and sellers to engage in the real estate market, ultimately leading to a more vibrant and sustainable market.

- Potential Fraud: Fraudulent activities remain a significant challenge for the real estate market and buyers as well as sellers alike are exposed to various risks. Real Estate transactions are vulnerable to various fraudulent activities such as mortgage fraud, title fraud and investment scams. They lead to major financial losses for the involved parties with long-lasting consequences. Mortgage fraud involves providing false or misleading information on mortgage application, which can lead to a lender providing a mortgage loan based on inaccurate information. Title fraud, for instance, involves falsifying property ownership documents, making it possible for an individual to sell to mortgage a property, which is not actually theirs. Investment frauds can involve a range of fraud tactics, such as offering unrealistically high returns on investments, collecting funds for non-existent projects, or misrepresenting property value. To minimize these risks in the market, implementing measures such as increasing public awareness, improving transparency, and strengthening regulations and enforcement mechanisms is quite essential. Therefore, the real estate market can become more secure, trustworthy, and reliable, thereby creating a more conducive environment for investment and sustainable growth.

- Legal Issues: Title disputes can be a common legal issue that arises when conflicting claims to property ownership occur. Legal action may be taken due to title disputes, which can cause substantial delays in the transaction process. Another legal challenge that can affect real estate transactions is zoning issues. This can be implemented which dictate the types of buildings or activities that are allowed on a property. Not adhering to zoning regulations results in fines, penalties, or even the destruction of a property. In addition, contract disputes can also arise during a real estate transaction, which result in additional costs, delays and even termination of the sale. Contract disputes can occur due to various reasons, such as disagreements over the terms of the contract, breach of contract, or misrepresentation.

These difficulties highlight the need for greater transparency, improved transaction processes, and more effective real estate regulation and oversight. Addressing these challenges will make the real estate market more accessible and efficient for both buyers and sellers.

1.3. GIS and Blockchain

Geographic Information System: It is a computer-based system that allows users to capture, store, manipulate, analyze, and visualize geographical or spatial data. This technology is used to create maps, analyze data, and solve real-world problems in a variety of fields, including urban planning, environmental management, public health, and natural resource management. GIS works by integrating data from various sources, such as satellite imagery, aerial photographs, and demographic and economic data, into a single digital platform. This data can be analyzed and visualized using various tools and techniques, such as spatial analysis and modeling, which allow users to gain insights and make variety of informed decisions based on geographic data.

Blockchain: is a digital ledger technology that allows for secure and transparent transactions without the need for a central authority. It is a decentralized system that uses cryptography to ensure transaction security and tamper resistance. Each transaction in a blockchain network is validated and recorded on a digitally shared immutable ledger maintained by the network's nodes, or rather computers. This provides a high degree of transparency since every member of the network can see the transaction history on the system, and it also provides a high level of security because the ledger cannot be altered without the consensus of the network. Immutability refers to the fact that once a transaction is recorded on the blockchain, it cannot be changed or deleted. This is because each block on the blockchain contains a unique cryptographic hash that is based on the previous block in the chain and hence is used to chain them up. This makes it virtually impossible to change any transaction data without changing the entire blockchain, which would require the consensus of the node network. The distribution of the blockchain ensures that it is more secure and resistant to attack, as there is no central point of failure that can be targeted by attackers.

> In blockchain innovation, gas is an estimation unit for any of the computational work required to execute an exchange or smart contract on the blockchain. Gas fees are basically the charges that clients pay to miners to execute exchanges on the blockchain. The higher the gas cost or gas limit, the higher the transaction fee. Miners on the blockchain.

Blockchain technology has a wide range of applications, including in finance, supply chain management, and digital identity verification. It is a powerful tool for creating more efficient, and transparent systems for conducting transactions. By using blockchain technology, organizations can reduce costs, improve transparency, and increase the speed and security of their transactions.

Decentralized Application commonly referred to as a DApp, is a software application that operates on a decentralized network, such as a blockchain. Unlike traditional applications that are controlled by a single entity, a DApp is governed by a set of contracts also known as smart contracts, executed on Blockchain, allowing a more transparent system.

Ethereum is an example of decentralized Blockchain application Ethereum has its own cryptocurrency called Ether (ETH), which is used to pay for transactions and incentivize nodes to validate transactions on the network. Ethereum uses a proof-of-stake consensus algorithm, which means that nodes on the network must perform complex calculations to validate transactions and add them to the blockchain. Developers can build a wide range of DApp on the Ethereum platform, from financial services like decentralized exchanges and lending platforms, to gaming and social media applications. They are accessible to anyone with an internet connection.

1.4. Survey

We conducted a survey and deduced results from the opinion of public. The survey consisted of several questions to determine the level of difficulty in the whole process of buying and selling land, as well as the issues & plausible ways to rectify them.

The typical buying process of a property is through real estate agents voted by 90.7% of the people. Figure 1 to Figure 14 show survey results conducted using google docs and results are shown in graphical form.

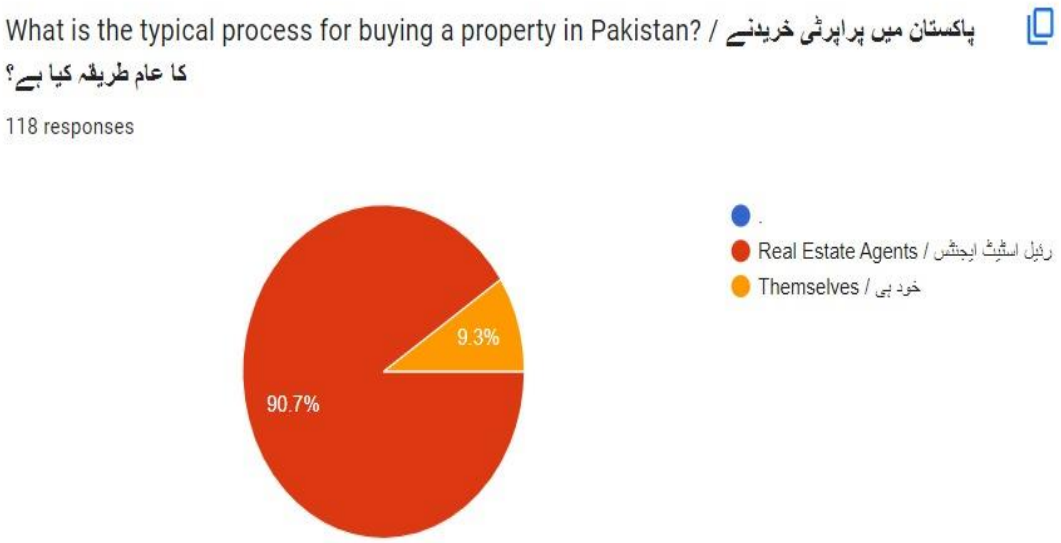


Figure 1: Survey (A)

When asked about the persistent challenges faced by buyers in the purchasing process 61.9% voted for 'Inaccurate or Incomplete Information about property' by seller followed by 'Complexity in Legal and Administrative process' by 55.9%. In addition, 50% voted to 'Title and Ownership disputes. There was an option to select as many as one thinks is the issue.

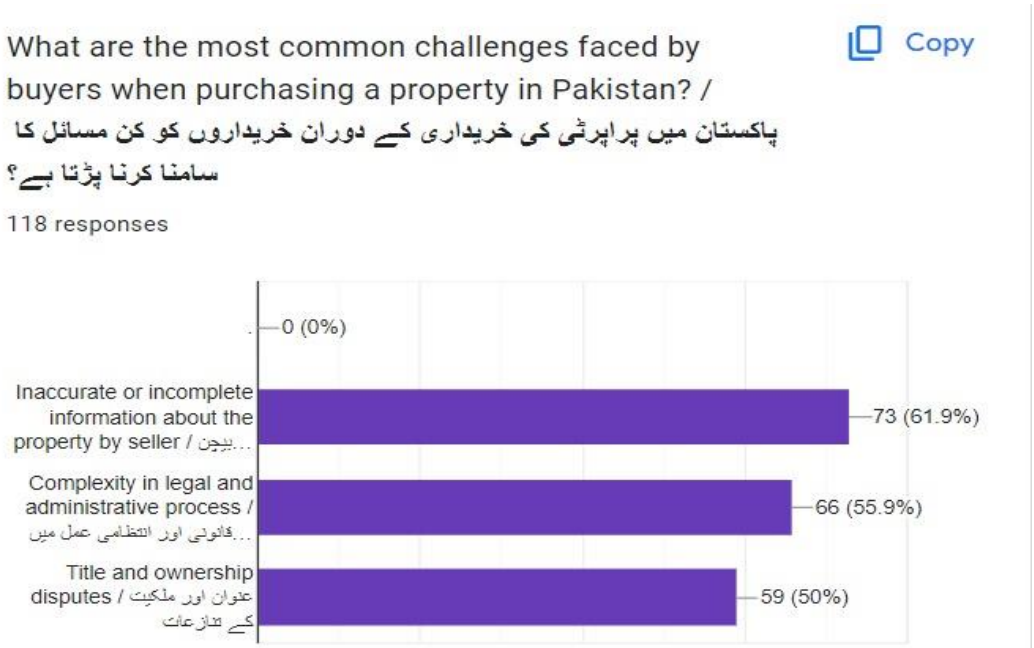


Figure 2: Survey (B)

With the chance of money recovery after a fraud, from least likely to most likely, 59.3% of the people voted for least likely.

Figure 3: Survey (C)

The process of Buying and Selling was mediocre for 33.9% of the people whereas, 22% voted it to be difficult.

Figure 4 : Survey (D)

With the level of importance from 1 to 5, 5 being the most important, 56.8% people voted for direct communication with the negotiating party as most important.

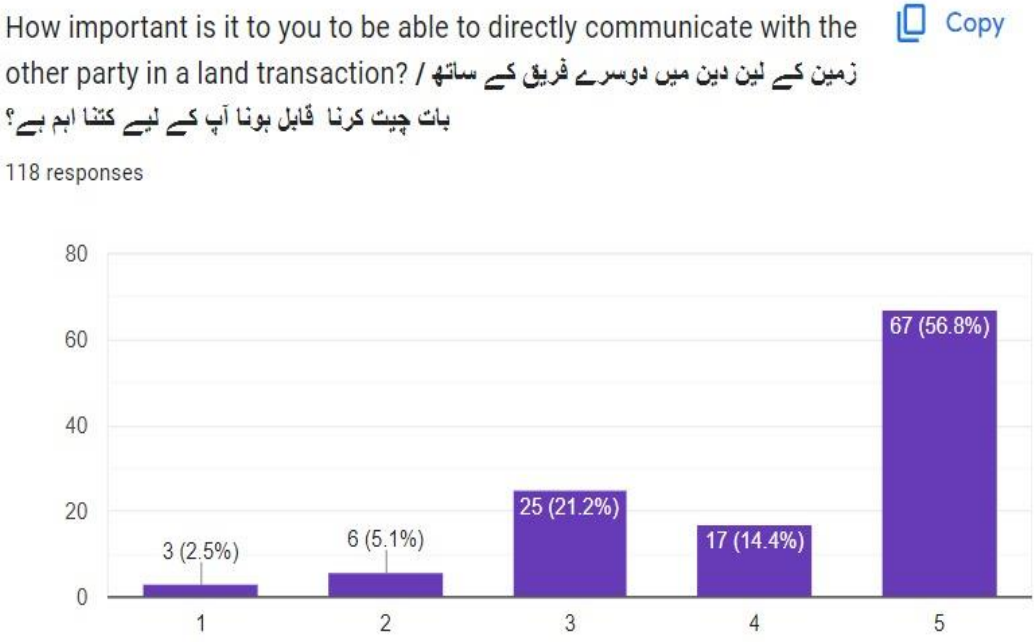


Figure 5: Survey (E)

84.7% were willing to pay a small fee for using a streamlined process.

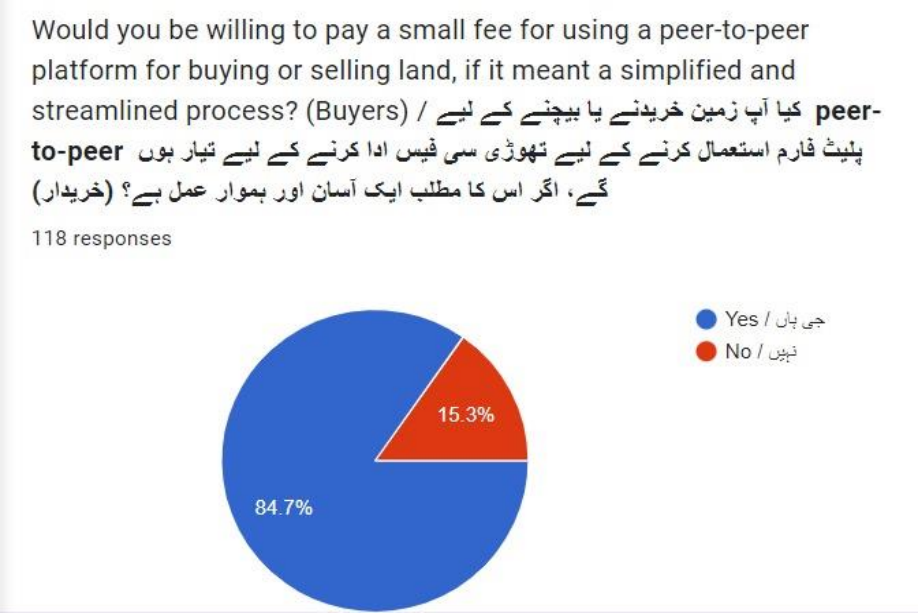


Figure 6 : Survey (F)

94.1% prefer to have full control of managing the property i.e., setting price, adding beneficiaries etc. as shown in the image below.

Figure 7 : Survey (G)

The current state of the real estate market got 42% neutral votes with 18.6% on low, on the scale of 1 to 5 being low to high.

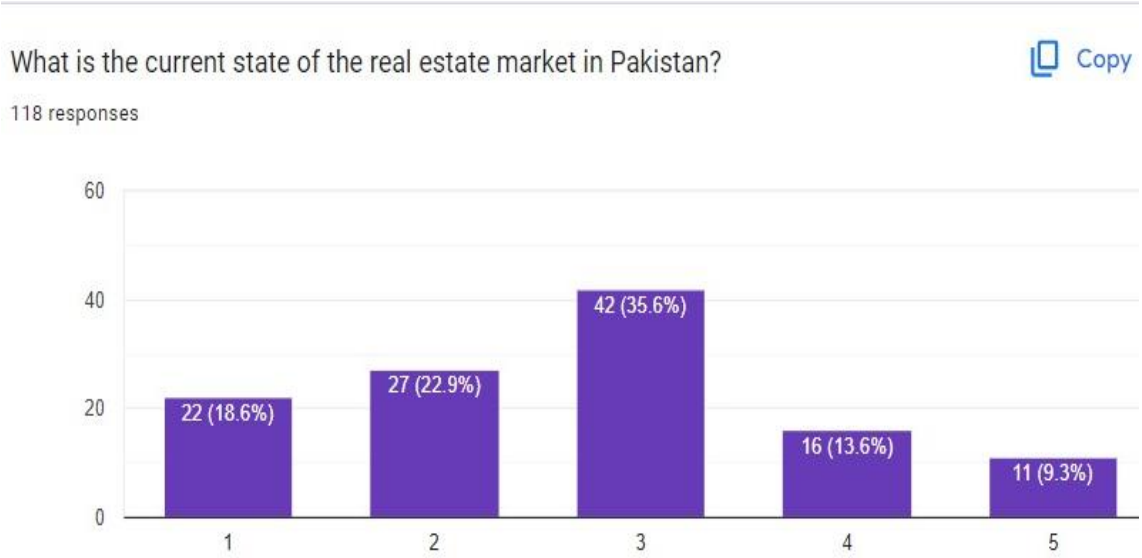


Figure 8 : Survey (H)

The following shows ‘Listing the property on real estate websites, social media platforms and the other online classifieds’ is the most favored one

Figure 9: Survey (I)

47.5% People voted 3 to 5 months to property sale.

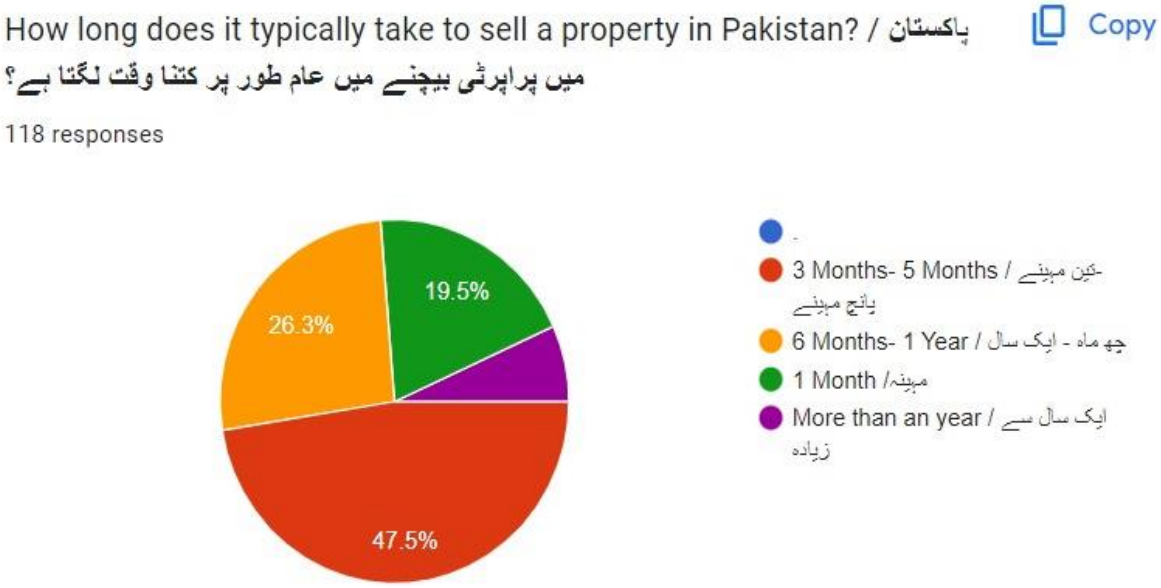


Figure 10: Survey (J)

Many people voted for low rate of security in real estate

Figure 11: Survey (K)

Fake property listings reported as the most common scam according to 61.9%.

Figure 12: Survey (L)

When asked about the role of preventing scams in the real estate market, 75.4% of respondents indicated that verifying property documents is the most effective solution.

Figure 13: Survey (M)

Interestingly, filing a lawsuit also received a significant percentage of responses, indicating that it is viewed as a viable option by the public.

Figure 14: Survey (N)

1.5. Scope

GIS & Blockchain Technology are two separate technologies, but they can be used together to create innovative solutions. This integration can be useful in various industries, such as real estate, where property records can be securely stored on a blockchain network and accessed by authorized parties in a transparent manner. This collaboration of GIS and blockchain has a potential to change the old existing system and provide a securer and more transparent system. This integration can also result in managing geospatial data with various potential applications in different industries. A decentralized application (DApp) for peer-to-peer buying and selling of land using GIS and blockchain technology could offer all these benefits.

Here is a possible outline of how such a DApp could work:

- User registration: Users would need to create an account on the DApp and provide some basic information about themselves, such as their name, CNIC ID, email address, and location (Ølnes, 2017).

- Land registration: Landowners would need to register their land on the DApp by providing details such as the location, size, and boundaries of their land. This information could be uploaded using GIS (Geographic Information System) technology, which would help to accurately map the land and prevent any disputes over its boundaries.

- Smart contracts: Once the land is registered on the DApp, smart contracts could be used to automate the buying and selling process (Karamitsos, 2018). These contracts would include the terms and conditions of the transaction, such as the price of the land, the payment method, and the transfer of ownership. They would be programmed to execute automatically once certain conditions are met, such as the transfer of funds.

- Tokens could be used in the DApp for peer-to-peer buying and selling of land using GIS and blockchain technology to represent ownership, enable fractional ownership in the decentralized arbitration process (Bhanushali, 2022). This would enhance the efficiency, transparency, and security of real estate transactions, while also creating new investment opportunities.

- Payment: Payments for the land could be made using cryptocurrency, which would provide a secure and transparent method of payment. Once the payment is made, the smart contract would automatically transfer ownership of the land to the buyer.

- Verification: To ensure the authenticity of the land and prevent any fraudulent transactions, the DApp could use blockchain technology to create a digital record of each. Blockchain technology would create an immutable and transparent record of each transaction, ensuring authenticity (Karamitsos, 2018). This record would be immutable and transparent, allowing anyone to verify the ownership of the land.

- Dispute resolution: In the event of a dispute between the buyer and the seller, the DApp could use a decentralized dispute resolution mechanism to resolve the issue. This would help to ensure that disputes are resolved fairly and transparently (Wouda, 2019). A DApp for peer-to-peer buying and selling of land using GIS and blockchain technology could offer many benefits, including increased transparency, security, and efficiency in the land transaction process. Integrating both technologies would also help to reduce fraud and ensure that land transactions are conducted fairly and transparently (Saari, 2022), (Bhanushali, 2022) (Ølnes, 2017).

1.6. Objective

The objective of our application is to create a decentralized application that operates on the blockchain network. The aim of the application is twofold: firstly, it seeks to provide a platform for peer-to-peer buying and selling, and secondly, it aims to maintain land records on the blockchain.

The first objective of the application is to provide a platform for peer-to-peer buying and selling that is facilitated by blockchain technology. This would allow users to engage in transactions without the need for centralized authority as we have many time mentioned previously, which would make the process faster, securer, and efficient. By using blockchain technology, the application would be able to provide a transparent and tamper-proof record of all transactions done, which would help to prevent fraud and ensure the integrity of the buying and selling process.

The second objective of the application is to maintain land records on the blockchain. They would be protected from unauthorized changes and could be easily accessed by authorized parties. By using blockchain technology, the application would be able to provide a transparent and tamper-proof record of all land records, which would help to prevent fraud and ensure the accuracy of land records.

1.7. Literature Review

Blockchain technology has emerged as a promising solution for various industries, including real estate. In the real estate industry, blockchain technology can enhance transparency, security, and efficiency by creating a tamper-proof and decentralized database that stores all transactions. This paper aims to review the literature on blockchain technology and its potential applications in the real estate industry.

According to Thota (2019), blockchain technology can enable secure and transparent property transactions by eliminating the need for intermediaries, such as banks and lawyers, and reducing the risk of fraud and errors. Smart contracts can be used to automate a number of procedures, including payment processing and property transfer, which cuts down on the time and expense involved in these transactions. In addition, the adoption of blockchain technology can enhance the management and tracking of property titles, deeds, and other critical documents, guaranteeing that they are safe, impenetrable, and easy to access. He also noted that Blockchain technology can make fractional ownership and investing in real estate properties easier, enabling people to buy high-value homes without having to put up a lot of money up front. This could increase investment possibilities and real estate market liquidity. Additionally, blockchain technology can increase the efficiency of property management by offering automated and transparent systems for rent collection, maintenance, and repair. However, the implementation of blockchain technology in the real estate industry is not without challenges. Thota (2019) identified several challenges, such as legal and regulatory barriers, technological infrastructure, and interoperability with existing systems. Moreover, the adoption of blockchain technology in the real estate industry requires a change in mindset and culture, as well as education and training on the use of this technology. In conclusion, blockchain technology has the power to revolutionize the real estate sector by improving accessibility, security, efficiency, and transparency. However, in order to successfully apply blockchain technology in the real estate sector, a number of obstacles must be overcome. To do this, multiple stakeholders—including industry participants, decision-makers, and technology providers—must work together.

According to the paper "Blockchain in Real Estate Sector: Benefits and Challenges" by Muhammad Umer Shabbir, the real estate industry is undergoing a digital transformation, and blockchain technology has the potential to revolutionize this industry by enhancing transparency, efficiency, and security. This paper provides a literature review of the benefits and (Karamitsos, 2018) possible difficulties of using blockchain technology in real estate. According to Mohanty et al. (2018), by building a decentralized database that stores all transactions in an immutable and tamper-proof manner, blockchain technology can improve the security and transparency of real estate transactions. By doing so, the necessity for intermediaries is removed, and the likelihood of fraud and mistakes is decreased. Additionally, the usage of smart contracts can automate and streamline a number of procedures, including payment processing and property transfer, cutting down on the time and expenses involved in these transactions.

Furthermore, Singh et al. (2019) noted that blockchain technology can also facilitate fractional ownership and investment in real estate properties, allowing individuals to invest in high-value properties without the need for large amounts of capital. This could create new investment opportunities and boost market liquidity for real estate. Moreover, according to the literature, blockchain technology can help handle property data and documents like titles, deeds, and leases. These documents can be safely and effectively viewed and validated by numerous parties by being stored on a blockchain, which lowers the possibility of mistakes and disputes. However, the adoption of blockchain technology in the real estate industry also presents several challenges. These include legal and regulatory barriers, lack of standardization, technological infrastructure, and interoperability with existing systems (Shabbir, 2019). Additionally, the increasing cost and complexity of implementing blockchain technology can be a significant barrier to adoption. In conclusion, blockchain technology has the power to revolutionize the real estate sector by improving accessibility, security, efficiency, and transparency. The implementation of blockchain technology in the real estate industry holds a number of hurdles that one must overcome in order to take these advantages.

In the paper "Blockchain for Real Estate" by Jyotsna Yarlagadda and Keerthi Gampala, the authors wrote that the real estate industry has been undergoing a digital transformation in recent years, and blockchain technology has emerged as a potential solution to enhance transparency, security, and efficiency in the industry. Yarlagadda and Gampala (2021) conducted a literature review on the potential applications of blockchain technology in the real estate industry. The authors noted that by establishing a decentralized ledger that preserves all transactions in an unchangeable and tamper-proof manner, blockchain technology

can simplify real estate transactions. By doing so, the necessity for intermediaries is removed, and the likelihood of fraud and mistakes is decreased. Additionally, the usage of smart contracts can automate and streamline a number of procedures, including payment processing and property transfer, cutting down on the time and expenses involved in these transactions. The authors pointed out that fractional ownership and investment in real estate properties can be made easier by blockchain technology, enabling people to make investments in high-value properties without having to spend a lot of money. This may create more investment options and boost market liquidity for real estate. Moreover, the authors highlighted that blockchain technology can improve the management of property data and documents, such as titles, deeds, and leases. By storing these documents on a blockchain, they can be efficiently and securely accessed and verified by multiple parties. This can reduce the risk of errors and disputes and improve the overall efficiency of the real estate industry. However, the authors also noted that the implementation of blockchain technology in the real estate industry faces several barriers, such as technological limitation, legal and regulatory barriers, technological infrastructure, and interoperability with existing systems. Therefore, the successful adoption of blockchain technology in the real estate industry will require collaboration among different industry stakeholders and the development of standardized protocols and frameworks.

In conclusion, they suggested that through improving transparency, security, efficiency, and accessibility, blockchain technology has the ability to completely change the real estate sector. But more work needs to be done in terms of research and development to get beyond the difficulties that come with using blockchain technology in the real estate sector.

In the paper "Blockchain technology in commercial real estate transactions" by Hugo Pieter Wouda and Raymond Opdenakker, the authors believe that Blockchain technology has been gaining attention as a potential solution to improve the commercial real estate industry by enhancing transparency, security, and efficiency. Wouda and Opdenakker (2019) conducted a literature review on the potential applications of blockchain technology in commercial real estate transactions.

According to the authors, blockchain technology can offer a safe and tamper-proof platform for the storage and transfer of real estate assets, doing away with the need for intermediaries and lowering the likelihood of fraud and mistakes. Additionally, the use of smart contracts can automate and streamline a number of

procedures, including the sale of properties and the collection of rent, cutting down on the time and expense involved in these transactions.

The authors also emphasized that blockchain technology can make fractional ownership and investment in commercial real estate holdings easier, enabling private investors and institutions to buy high-value buildings without putting up a lot of money. This may create more investment possibilities and boost market liquidity for commercial real estate.

The authors also noted that the management of property data and documents, such as lease agreements, property titles, and ownership records, could be enhanced by the use of blockchain technology. Everyone can safely and quickly access and validate those documents by numerous parties after they are uploading on blockchain. This lowers the possibility of mistakes and disagreements and boosts the overall effectiveness of the commercial real estate sector.

The authors also pointed out constraints and limitations, including legal and regulatory constraints, infrastructure requirements, and the need for standardization and interoperability with current systems. Therefore, cooperation among industry stakeholders and the creation of standardized protocols and frameworks will be necessary for the successful application of blockchain technology in commercial real estate transactions. In conclusion, Wouda and Opdenakker (2019) suggested that blockchain technology has the potential to transform the commercial real estate industry by enhancing transparency, security, efficiency, and accessibility. However, further research and development are needed to overcome the challenges associated with implementing blockchain technology in commercial real estate transactions.

In an article, Sheraz Ahsan emphasizes that for centuries, the management of land ownership has been based on the registration of numbers and the preparation of maps based on land surveys. The British government introduced this system in the subcontinent almost 200 years ago for land ownership, along with legislation and administrative procedures. Although that system was not primarily built for land ownerships and transfer, the applied commissioner system from the British era is still in force today. Under this system, there were two registers for land ownership numbers, which included details of the owner's personal details, sections, total area, lease, and irrigation facilities. After the partition of subcontinent, Pakistan inherited the British system of land ownership, which remains largely unchanged to this day. That system was not outdated, any land transfer was not possible without bribery. The Patwari maintained official records in his private possession and derived personal benefits from it. In 2006, the World Bank

declared this system highly outdated, inaccessible, weak, inefficient, and ineffective in its report. According to an international study conducted in 2013, 90% of the population affected by this system wants it reformed. In 2016, Punjab government, Pakistan initiated the computerization of land records process in collaboration with World Bank. However, the mapping component of the system is still not clear. All paper-based records were stored in computers for land computerization, and the errors are still there because they were removed by the current corrupt system. Now the Land record centers have been formed, and all the information can be obtained under one roof.

The research was conducted by author for NUST University, Pakistan revealed some flaws in computerized system. The study was conducted in location Chak No. 154 ML, Tehsil Kot Adu, District Muzaffargarh. The result showed that 8 acres of land were recorded on paper while no physical existence of the land was found on the ground. Modern mapping techniques were also full of errors and lacking technical skills. In conclusion, the current land management is outdated and corrupted, and there is dire need for a secure, transparent, and immutable system to be implemented.

According to another article “Integrated geospatial evaluation of manual cadastral mapping: a case study of Pakistan M. S. Ahsan, E. Hussain & Z. Ali”, The authors begin by providing background information on the importance of cadastral mapping and the challenges faced in developing countries like Pakistan. They then discuss the use of geospatial technology and its potential to improve the accuracy and quality of cadastral maps. The study focuses on the manual cadastral maps of two villages in Pakistan and compares them to the digital cadastral maps developed through the use of geospatial technology.

The literature review in the article covers various topics related to cadastral mapping, geospatial technology, and their integration. The writers talk about how cadastral mapping has evolved over time and how important it is for managing and administering land. They also give an outline of the difficulties encountered in putting into place accurate and effective cadastral systems in developing nations.

With an emphasis on the benefits of digital mapping over manual mapping, the use of geospatial technology in cadastral mapping is also reviewed. The authors address the difficulties in adopting these systems in underdeveloped nations while also discussing how geospatial technology may enhance the precision and effectiveness of cadastral mapping.

The literature review also covers the concept of geospatial evaluation and its importance in assessing the accuracy and quality of cadastral maps. The authors discuss various methods of geospatial evaluation, including visual inspection, statistical analysis, and error analysis. Overall, the authors provide a comprehensive overview of the current state of cadastral mapping and geospatial technology, and their integration. The authors highlight the importance of accurate and efficient cadastral mapping systems and the potential for geospatial technology to improve these systems in developing countries like Pakistan.

MATERIALS AND METHODS

2.1. Design and Analysis

We moved forward with the project's design requirements after completing a thorough analysis of the various blockchain technologies and related architectures necessary for the effective deployment of a decentralized application (DApp). The system environment, web architecture, and application-specific architecture are all included in this. Unified Modeling Language (UML) Diagrams will be the emphasis of the following section of this chapter, which will explain the fundamental principles and logic of our DApp.

2.1.1. System Environment

All of the technologies used in our project—including servers, databases, internal and external applications, application engines, and operating systems—are distributed under an Open-Source Community License.

2.1.1.1. Windows

Windows was the operating system used during the project's development, although the finished product (the website/DApp) can be accessed on any platform. Windows is very popular and provides a user-friendly interface in addition to interoperability with a variety of tools and software.

2.1.2. Web Architecture

Our project's web architecture is a three-tiered architecture with certain logical tier adjustments to support blockchain processing and queries. The stages of the web architecture are as follows:

1. Presentation Tier
2. Logical Tier
3. Data Tier

1. **Presentation Tier**

The user-facing interface that shows the customer the finished result is known as the presentation tier. Any web browser on any operating system can be used to access the web application. To access the web portal, utilize a popular web browser such as Firefox, Chrome, Opera, or Internet Explorer.

2. **Logical Tier**

Our decentralized application's logical tier is essential for providing a seamless user experience. It acts as a link between the blockchain network and the front end. Our program uses the Ethereum blockchain as its underlying technology, making it possible to execute smart contracts and securely handle digital land tokens. These Solidity-coded smart contracts streamline the purchasing and selling of tokens. When criteria are satisfied, they are intended to carry out predetermined actions.

We have created several APIs to enable seamless communication between the front-end and back end. Users are given the ability to carry out crucial operations including buying and selling digital land tokens, keeping track of transactions in real-time, and viewing their transaction history thanks to these APIs. We guarantee a user-friendly interface that enables seamless interaction with the features of our program by integrating these APIs.

Additionally, before publishing the smart contracts on the primary Ethereum network, we test and debug them using Ganache, a private blockchain for Ethereum development. This enables us to confirm the proper operation and dependability of the contracts. We rely on MetaMask, a digital wallet that gives customers a secure and easy-to-use interface for buying and selling digital land tokens, for secure interaction with the Ethereum network. Our logical tier offers a seamless and secure user experience across the decentralized application with the help of these elements.

3. **Data Tier**

The data tier of our decentralized program is in charge of handling both spatial and non-spatial data pertaining to virtual land tokens. For the purpose of visualizing geographic information and generating polygons that represent land plots, we use Google Earth Pro. The storage and management of the plot data is handled by PostgreSQL, an open-source relational database management system that supports a variety of data types, including JSON. We utilize Prisma, an object-relational mapping (ORM) tool that creates a type-safe client library, to communicate with the database in a type-safe and user-friendly way.

Our data tier uses PostgreSQL, Prisma, and Google Earth Pro together to efficiently manage and track data related to virtual land tokens. This design ensures effective data storage, retrieval, and visualization, which helps our decentralized application run smoothly.

2.1.3. System Architecture

Application architecture is based on set tools and libraries/ packages on which our DApp is build. Following description figure is provided that shows basic understanding of each component that were used for successful development and implementation on DApp.

Figure 15 shows DApp components.

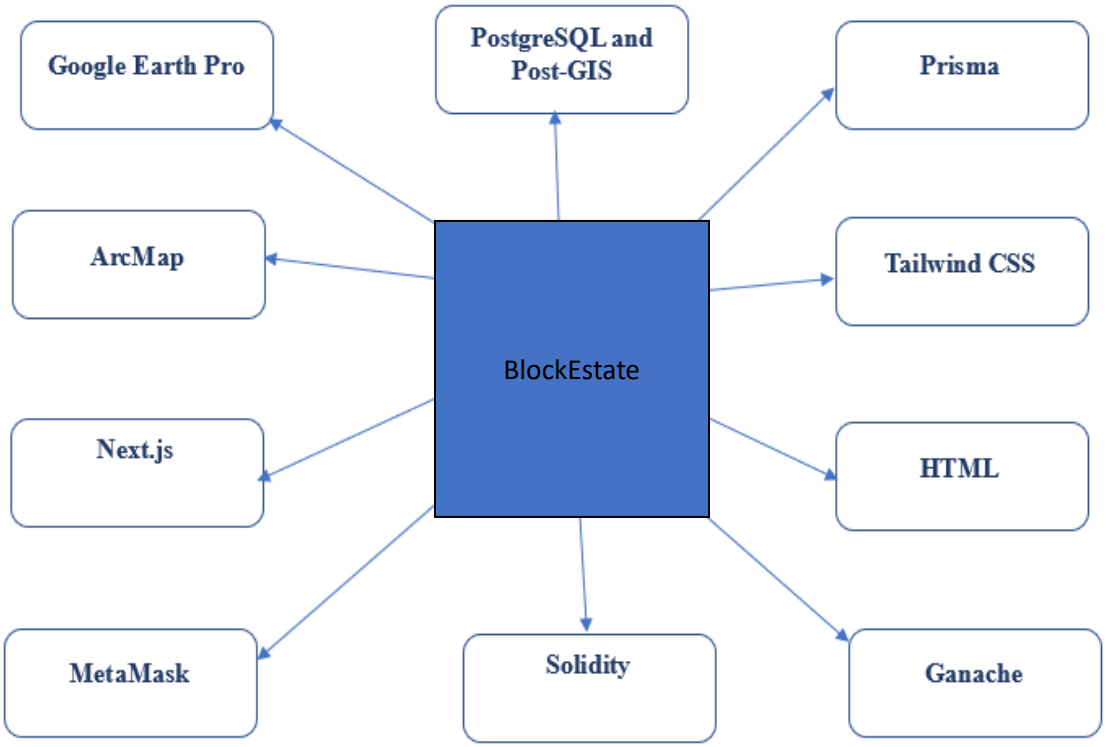


Figure 15: System Architecture

1. **Google Earth Pro:**

Online Earth Pro, a desktop program called Google Earth Pro is used for exploring and visualizing geographic data. In order to generate interactive maps and visualizations, it offers satellite imagery, 3D buildings, topography data, and other elements. We utilized it in our project to make polygons that symbolize shabby plots in Islamabad.

2. **ArcMap:**

The ArcGIS software suite includes Arc Map, which is used to create, manage, analyze, and visualize geographic data. Users can do spatial analysis, generate maps, and share the results with others. It is used with PostgreSQL because it converts KML files into shape files, and to cross check the accuracy and exact location.

3. **PostgreSQL and Post-GIS:**

Strong data administration and storage capabilities are provided by the sophisticated open-source relational database management system PostgreSQL. It has sophisticated capabilities like triggers, stored procedures, and full-text search and supports a large variety of data types, including JSON. The dummy plot data needed for the virtual land tokens is stored in databases created specifically for our project.

4. **Prisma:**

An object-relational mapping (ORM) tool called Prisma offers a type-safe and user-friendly approach to communicate with databases. Developers can access the database via a collection of highly typed APIs thanks to the type-safe client library it creates for the application.

5. **Tailwind CSS:**

. Tailwind CSS is a CSS framework that provides us with single-purpose utility classes, which are opinionated for the most part, and which help us design our web pages from right inside our markup or .js/ .jsx/ .ts/ .tsx files. In short is a utility-first CSS framework for rapidly building modern websites without ever leaving your HTML.

6. **Next.js:**

A well-liked React framework for creating server-side rendering (SSR) web applications is called Next.js. Out of the box, it offers a number of features like automatic code splitting, static site generation, and

improved performance. The project uses Next.js to render content on the server and run JavaScript on the client, giving users a quick and responsive experience.

7. **HTML:**

HTML is used to define the structure of components in complex web application

Solidity:

The Ethereum blockchain uses the high-level programming language Solidity to create smart contracts. It is used to write code that is executed on the Ethereum Virtual Machine (EVM), and is intended to be similar to JavaScript. It is utilized in this project to create smart contracts on Ethereum blockchain.

8. **MetaMask:**

For interacting with the Ethereum blockchain, utilize the MetaMask digital wallet. It is employed in our project to give users a safe and user-friendly interface for buying and selling digital land tokens.

9. **Ganache:**

A personal blockchain for Ethereum development, Ganache enables smart contract testing and debugging. It is utilized in our project to verify the smart contract's functionality prior to putting it into usage on the main Ethereum network. It is utilized for the project's local blockchain development and testing.

2.1.3 Activity Diagrams

Activities inside a software system are depicted using activity diagrams, commonly referred to as flowcharts. They help to describe the system's functionality and offer an outside perspective on the software operations.. Activity diagrams are a useful tool for business-process modeling because they provide information in a recognizable style.

These diagrams merely show how the actions in DApp "BlockEstate" are carried out. As a result of these diagrams, users can better understand the sequence of events that take place on the site for a given activity.

A small number of shapes joined together with arrows are used to create activity diagrams.

The most significant and prevalent shapes are:

• Actions are depicted by rounded rectangles from all four corners.

The choices are symbolized by diamonds.

• The split and end join of concurrent activities are represented by bars.

• The diagram's beginning is shown as a circle with shading.

• The conclusion is shown by a shaded circle surrounded.

Activity diagrams are shown from Figure 16 to Figure 21:

1. **Login**

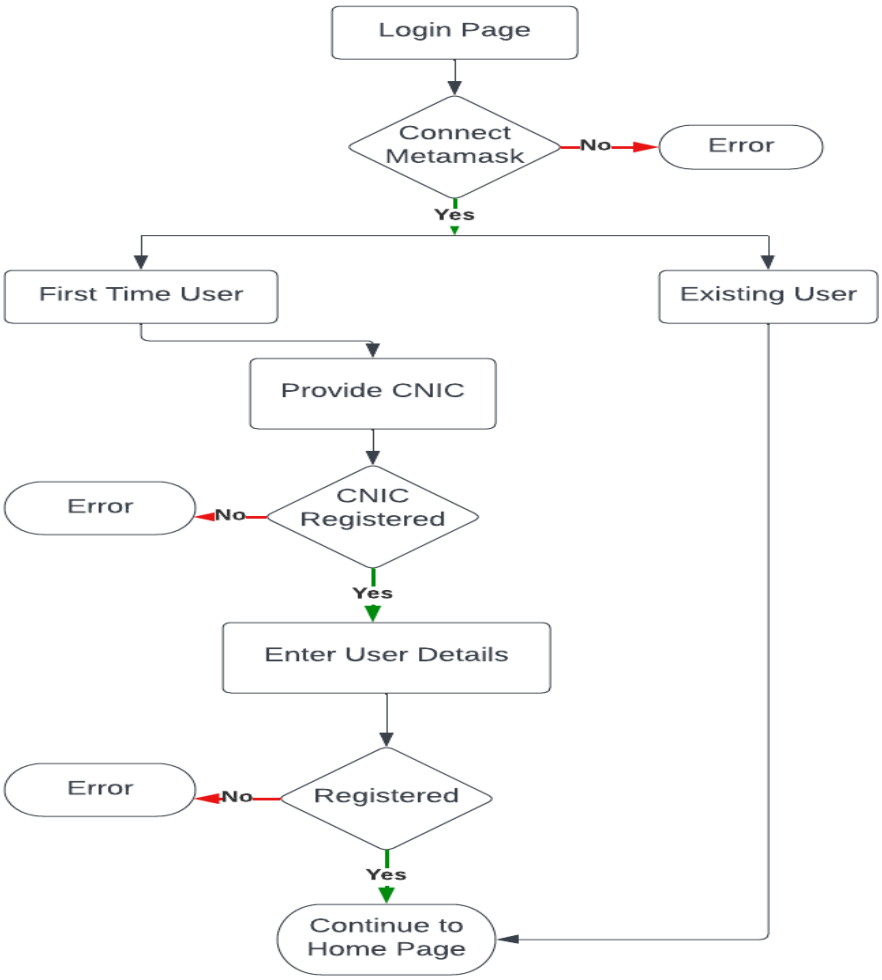


Figure 16: Login Activity Diagram

2. Log Out

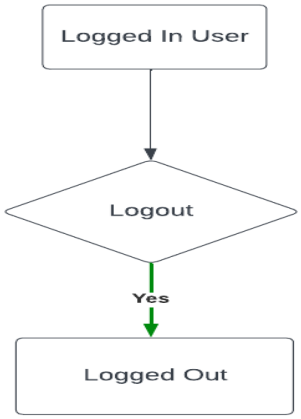


Figure 17: Logout Activity Diagram

3. Transaction

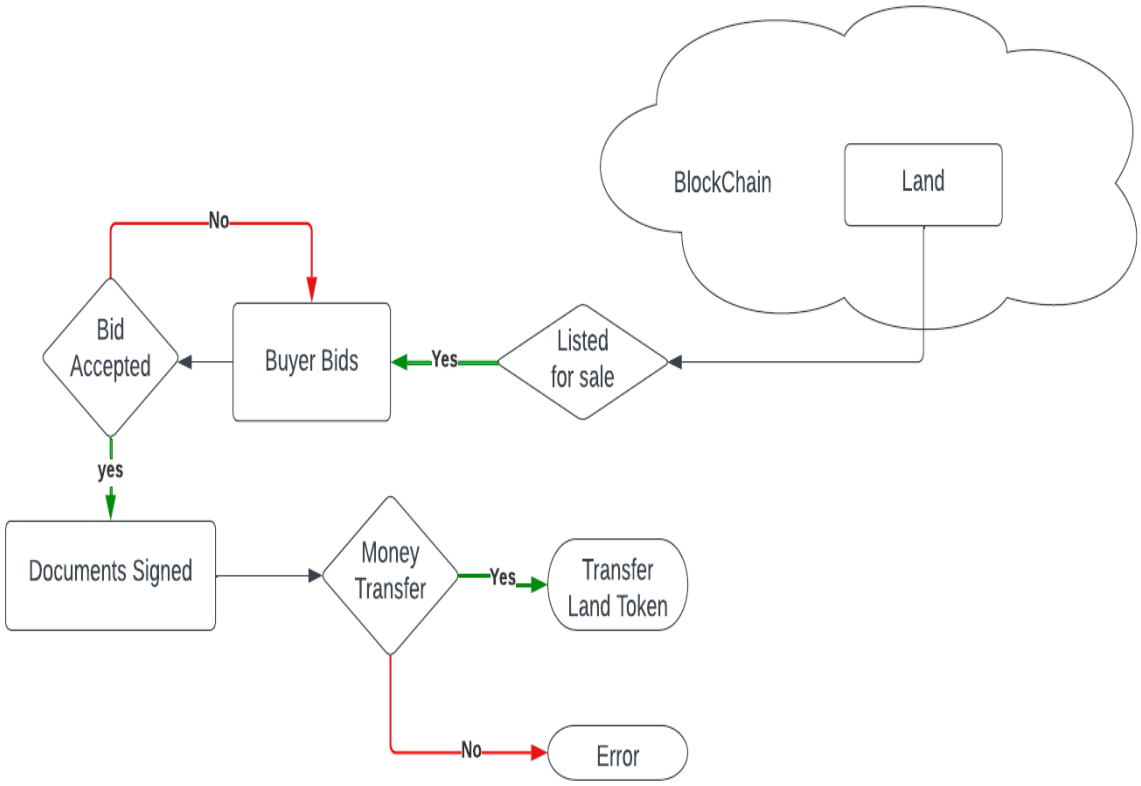


Figure 18: Transaction Activity Diagram

4. Verification

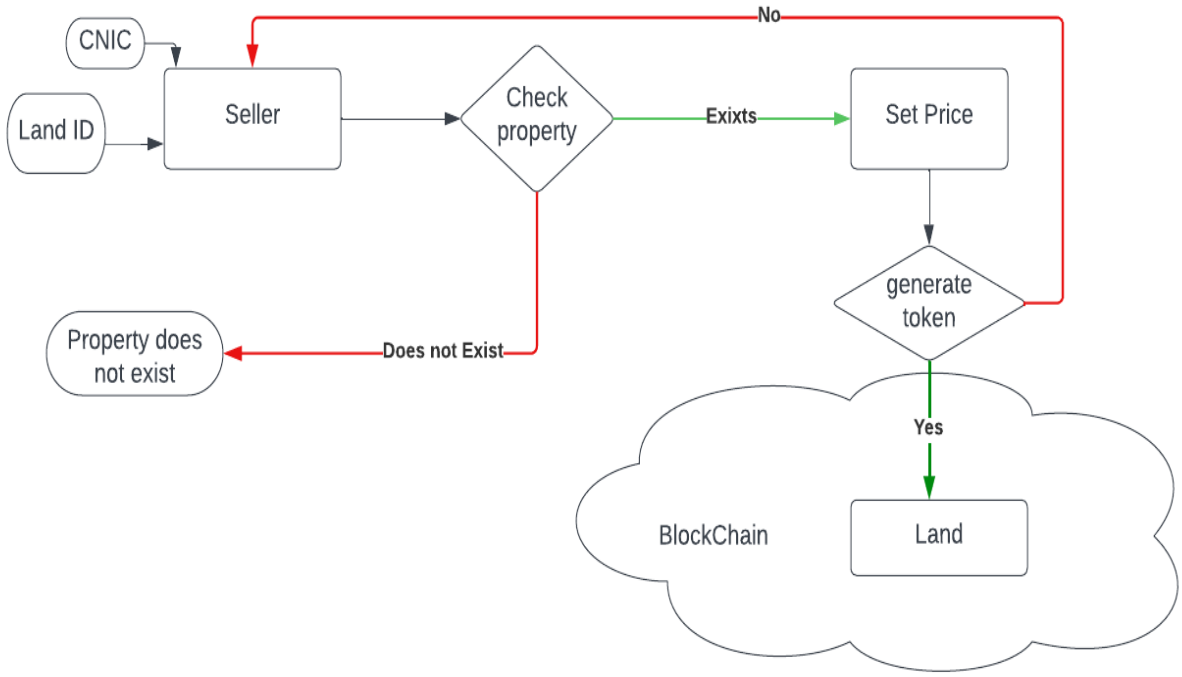


Figure 19: Verification Activity Diagram

5. Add Shared Owners

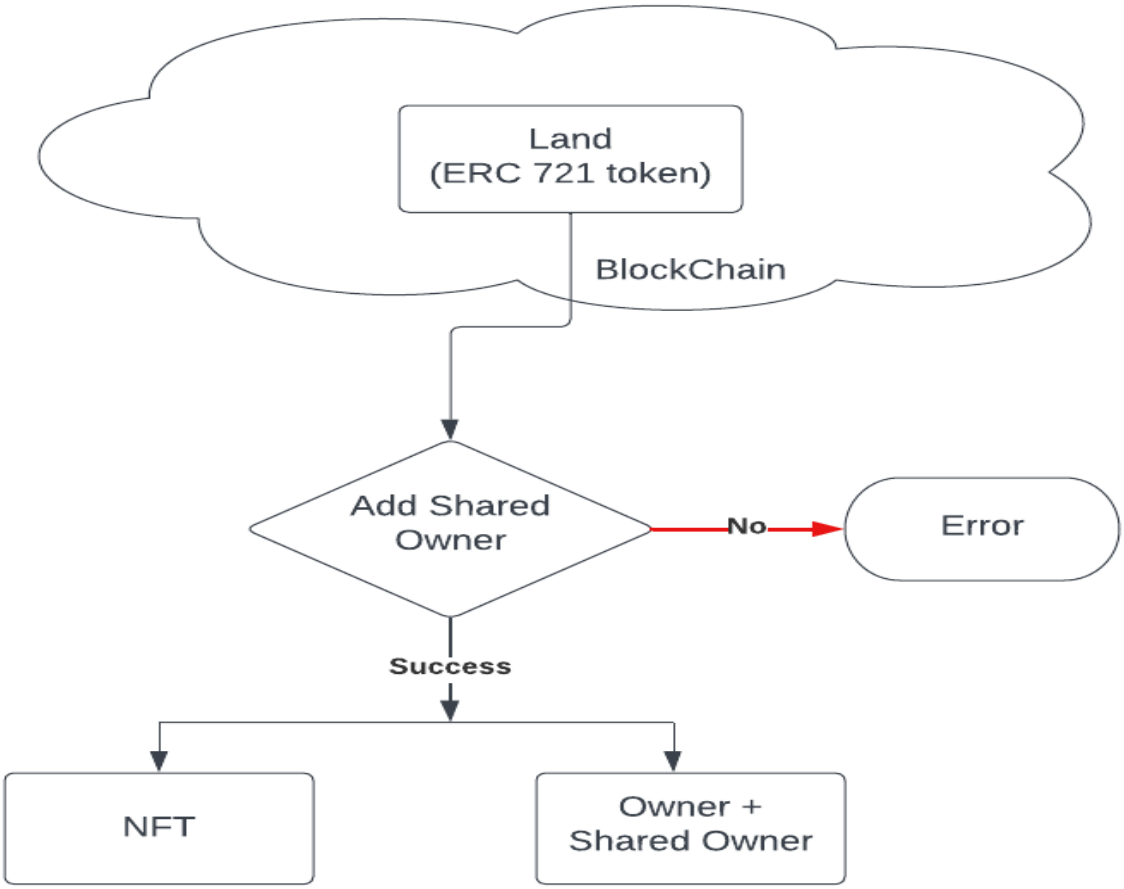


Figure 16: Add Shared Owners Activity Diagram

6. Remove Shared Owners

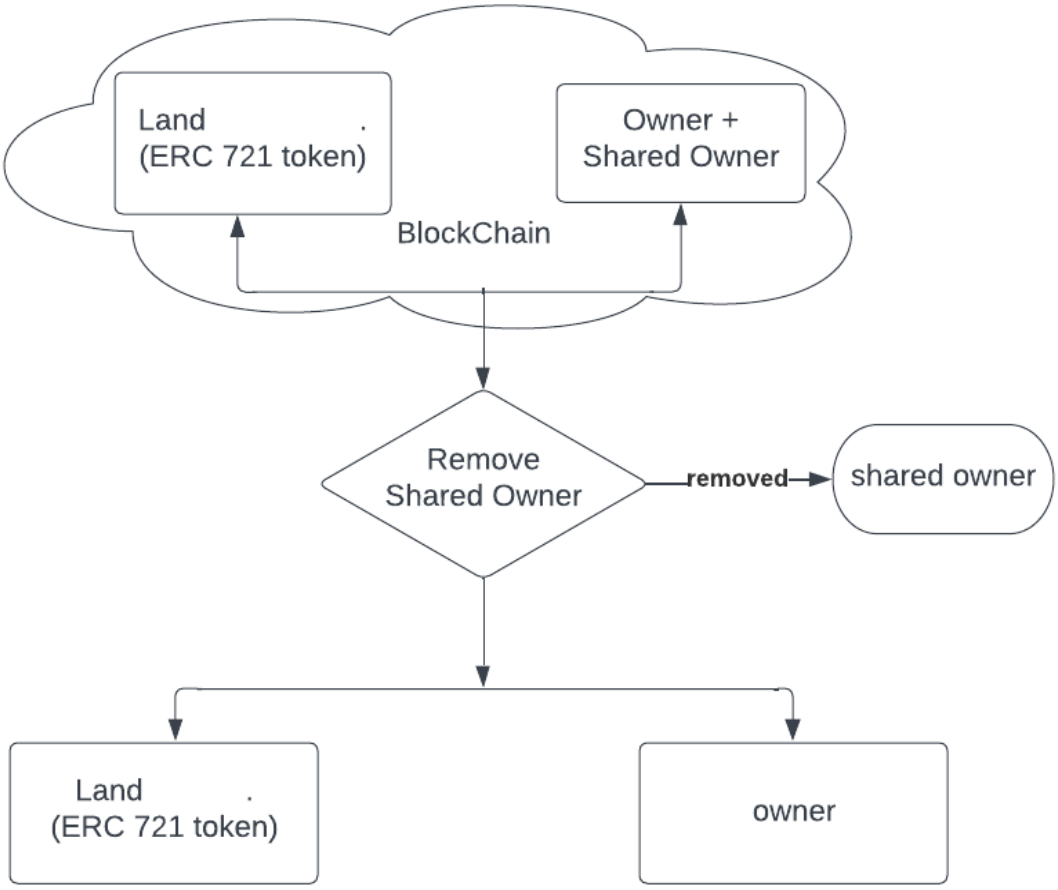


Figure 17: Remove Shared Owners Activity Diagram

2.2 Data Preparation and Database Setup

1. **Shapefile Creation:**

Selected a location in Islamabad, Pakistan, where building is not yet underway. Then we created polygons in Google Earth Pro, added columns that measure the area and length of area under polygon, and mention its types like residential and commercial with a KML file format. Then we used ArcMap to convert the KML files to shapefiles. To import the shape files we prepared in ArcMap, we created tables in PostgreSQL and installed the postgis shapefile import/export manager bundle extension. Then, to hold the data, we made tables.

2. **Database Creation:**

Then we have used PostgreSQL to store and handle the data acquired from the polygons and shapefiles. We were able to import the shapefiles made in ArcMap into PostgreSQL after creating tables and installing the postgis shapefile import/export manager bundle extension.

By creating the tables, we have established a system for efficiently managing the information. This process ensures that potential buyers have access to accurate and reliable information about the property they are interested to look.

We created four tables described as follows:

a. **Table: users:** We created this table to store the data of the user at the time they registered and updated it whenever they change the information on my account page. This table consist of 8 columns:
   "ID" is a default column in every table. It gives the table a primary key every time a new row is created. It stores integer values.
   "address" stores the Ethereum wallet address of every user.
   "CNIC" stores the CNIC of every user.
   "Name" stores the name of every user.
   "Email" stores the Full email of every user.
   "is_register" contains three values 0,1,2. 0 represents the user has only logged in. 1 represents the user has entered CNIC. 2 represents all the details filled in by the user and registration is complete.
   Flowchart in Figure 15 and Figure 16 show complete login process.

Figure 22 to Figure 31 show database tables and their Activity diagrams

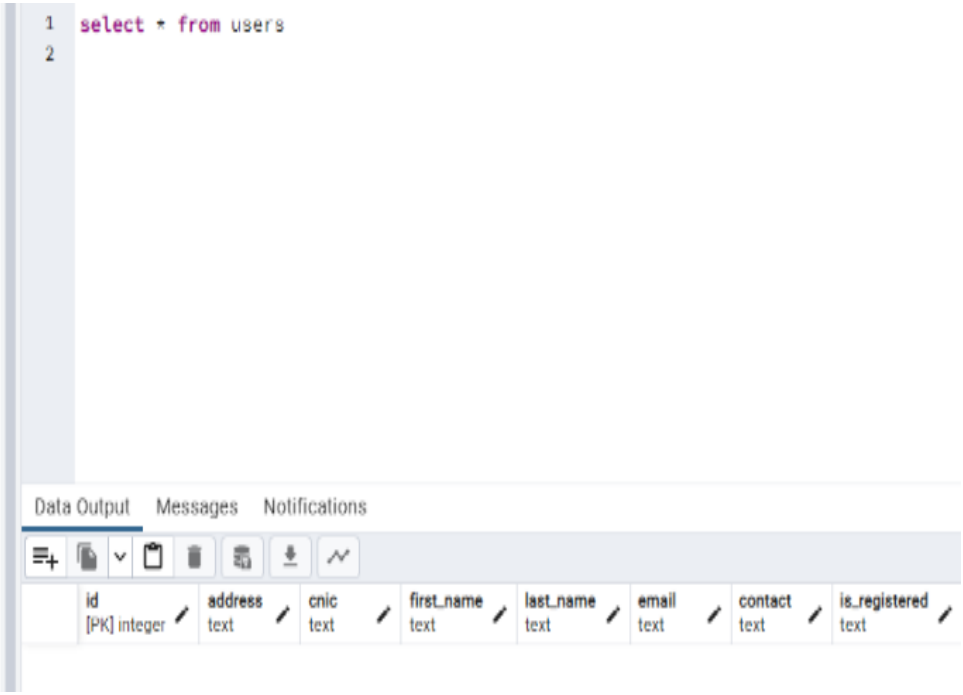


Figure 18: User Table Creation

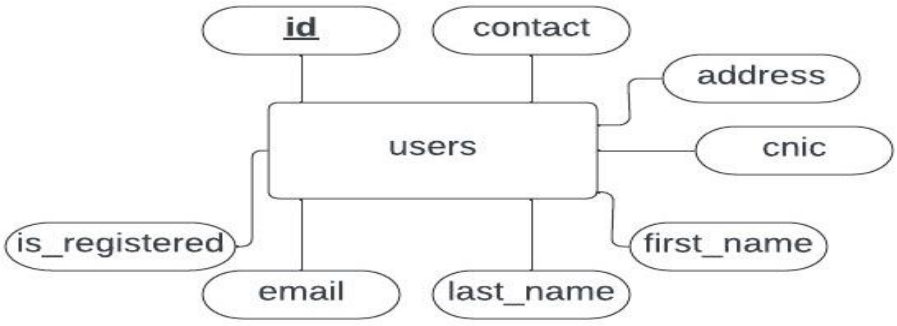


Figure 19: User Table Activity Diagram

b. **Table: sessions:** When a user logs in, a token is generated, this table is to store the login session details. This table consist of 5 columns:

“ID” is a default column in every table. It gives the tab;le a primary key every time a new row is created. It stores integer values.

“address” stores the wallet address of the user who logs in.

“session_id” stores a unique id every time a new token is generated.

“token” stores a unique token for every session.

“status” contains two values Expire and Active. If a session is ongoing it shows Active and if it ends then the value will be Expire.

Figure 17 and Figure 18 show further details.

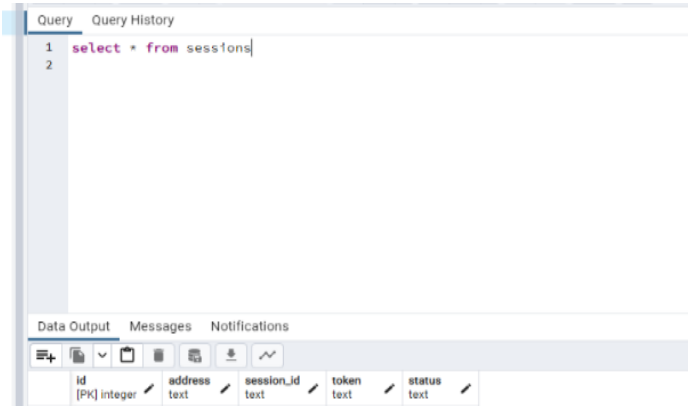


Figure 20: Session Table Creation

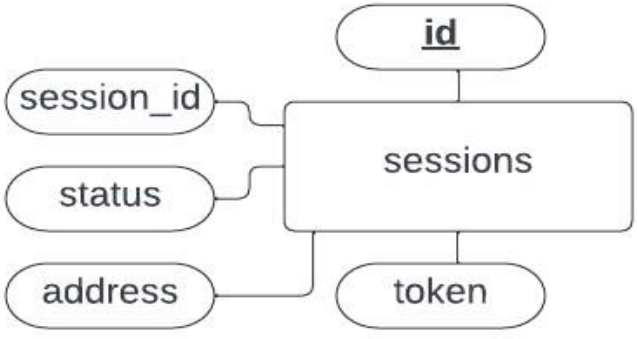


Figure 21: Session Table Activity Diagram

c. **Table: land_sale:** This table is connected to the marketplace. All the land details are added here.
“ID” is a default column in every table. It gives the table a primary key every time a new row is created. It stores integer values.
“land_id” stores the id of each land.
“address” stores the wallet address.
“usercnic” stores the CNIC.
“price” stores the land price and updates whenever the user changes the value.
“land_address” stores the address of the land.
“type” contains the two values "Commercial" and "Residential," it likely represents properties that are intended for commercial use and those that are intended for residential use.
“area” shows the area in m^2.

Figure 22: Land Sale Table Creation

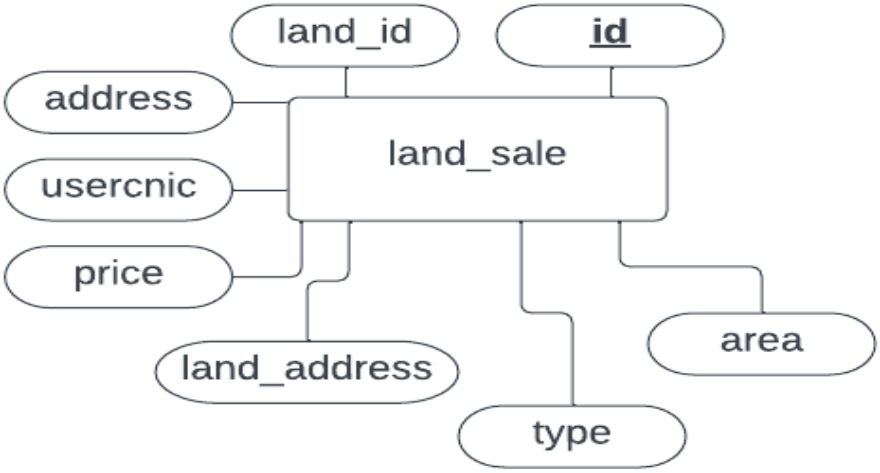


Figure 23: Land Sale Activity Diagram

d. Table: bid_request: This table stores the details related to the bids.

"ID" is a default column in every table. It gives the tab;le a primary key every time a new row is created. It stores integer values.

"land_id" stores the id of each land.

"seller_address" stores the seller's wallet address.

"buyer_address" stores the buyer's wallet address.

"ask_price" represents the price that is actually asked by the seller.

"bid_price" represents the price that is offered by the buyer.

"is_status" contains four values 0,1,2,3. 0 is the default value, 1 is when a bid is accepted and 2 is signed by both,3 is payment confirmed and land transferred.

"is_seller_signed" contains two values '0' and '1', 0 is no and 1 is yes.

"is_buyer_signed" contains two values '0' and '1', 0 is no and 1 is yes.

Figure 24: Bid Request Table Creation

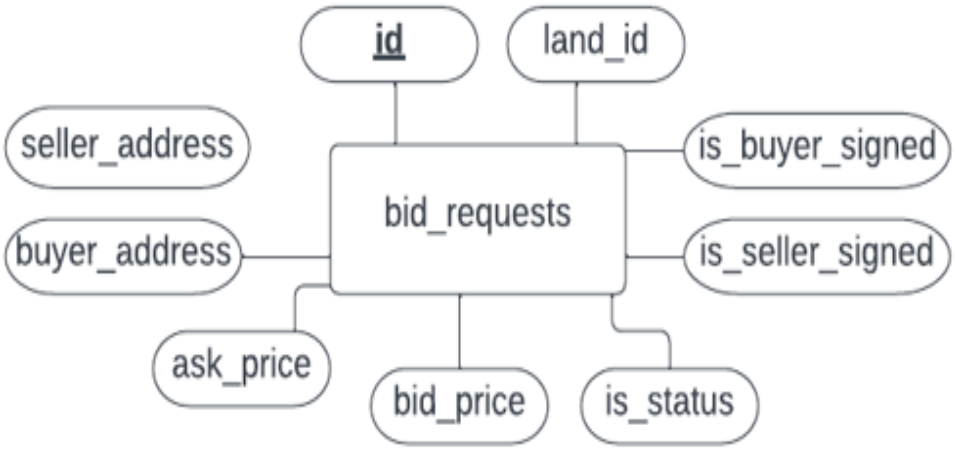


Figure 25: Bid Request Activity Diagram

e. **Table: plots:** It is to store the details of land and to store its geometry.

"ID" is a default column in every table. It gives the table a primary key every time a new row is created. It stores integer values.

"name" stores users name and updates when user change the name

"land_id" stores the id of each land.

"address" stores the wallet address.

"usercnic" stores the CNIC.

"price" stores the land price and updates whenever the user changes the value.

"land_address" stores the address of the land.

“geometry” stores the shape of the land in GeoJson format.

“type” contains the two values "Commercial" and "Residential," it likely represents properties that are intended for commercial use and those that are intended for residential use.

“area” shows the area in m^2.

“shape_leng” shows length of an area in meters.

gid [PK] integer	land_id double precision	name character varying (80)	type character varying (80)	status character varying (80)	usercnic character varying (80)	symbolid double precision	shape_leng numeric	shape_area numeric	geo geo
1	1	c1	Commercial	0	9040301451661	0	225.731	2963.690	01060000
2	6	6	c6	Commercial	0	9040301451661	0	103.636	647.920	01060000
3	4	4	c4	Commercial	0	9151412562772	0	102.496	635.519	01060000
4	9	9	r2	Residential	0	[null]	0	166.460	1714.510	01060000
5	10	10	r3	Residential	0	[null]	0	134.472	1101.663	01060000
6	12	12	r5	Residential	0	9151412562772	0	151.871	1392.017	01060000
7	13	13	r6	Residential	0	9040301451661	0	160.008	1562.239	01060000
8	11	11	r4	Residential	0	9040301451661	0	163.841	1659.869	01060000
9	3	3	c3	Commercial	0	[null]	0	181.976	2027.124	01060000
10	7	7	c7	Commercial	0	[null]	0	112.804	790.782	01060000
11	5	5	c5	Commercial	0	9151412562772	0	116.426	836.313	01060000
12	8	8	r1	Residential	0	9151412562772	0	159.403	1545.831	01060000
13	2	2	c2	Commercial	0	9040301451772	0	171.456	1737.216	01060000

Figure 26: Plots Table

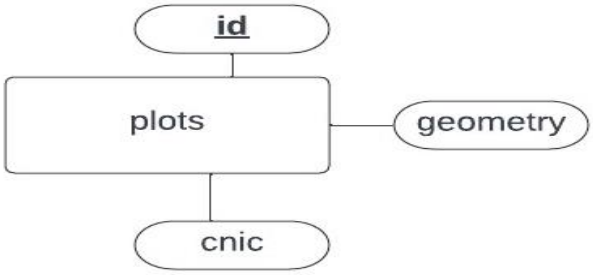


Figure 27: Plots Table Activity Diagram

2.3. Environment Setup for Smart Contracts on Localhost

1. Next App Initiation

Created a folder for the project, named BlockEstate.

Initialize a new npm project in the folder by running the command npm init.

2. Packages Installation:

Then we installed the packages/libraries.

Refer to Appendix B for the list of packages/libraries.

3. Tailwind CSS

Install Tailwind CSS by running the command:

npm install tailwindcss.

In style.css file, add the following code:

@tailwind base;
@tailwind components;
@tailwind utilities;

Now we can start building the "BlockEstate" app in Next.js using Tailwind CSS

4. Prisma

Using Prisma, we build a PostgreSQL database.

To install Prisma globally, we use npm by running the following command:

npm install prisma -g

To initialize a new Prisma project, we run the following command

prisma init

This will ask you to choose the kind of database you wish to use, create a new Prisma schema file (prisma/schemaPrisma), which you can use to specify your data model, and produce a client that you can use to communicate with your database..

5. Ganache

We downloaded Ganache for windows and created a new workspace. Then we connected the truffle-config.js file to the workspace.

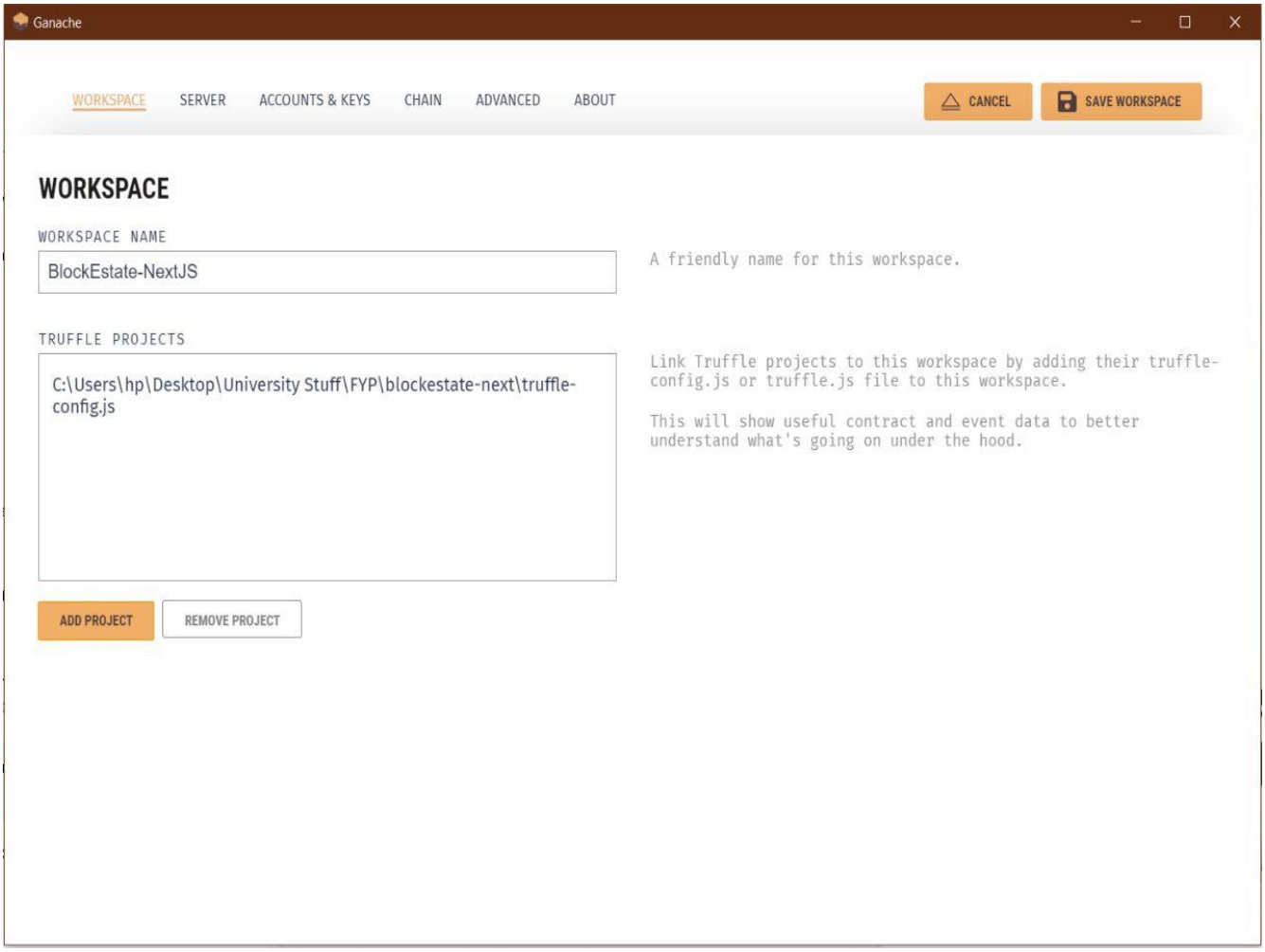


Figure 28: Ganache Workspace Setup

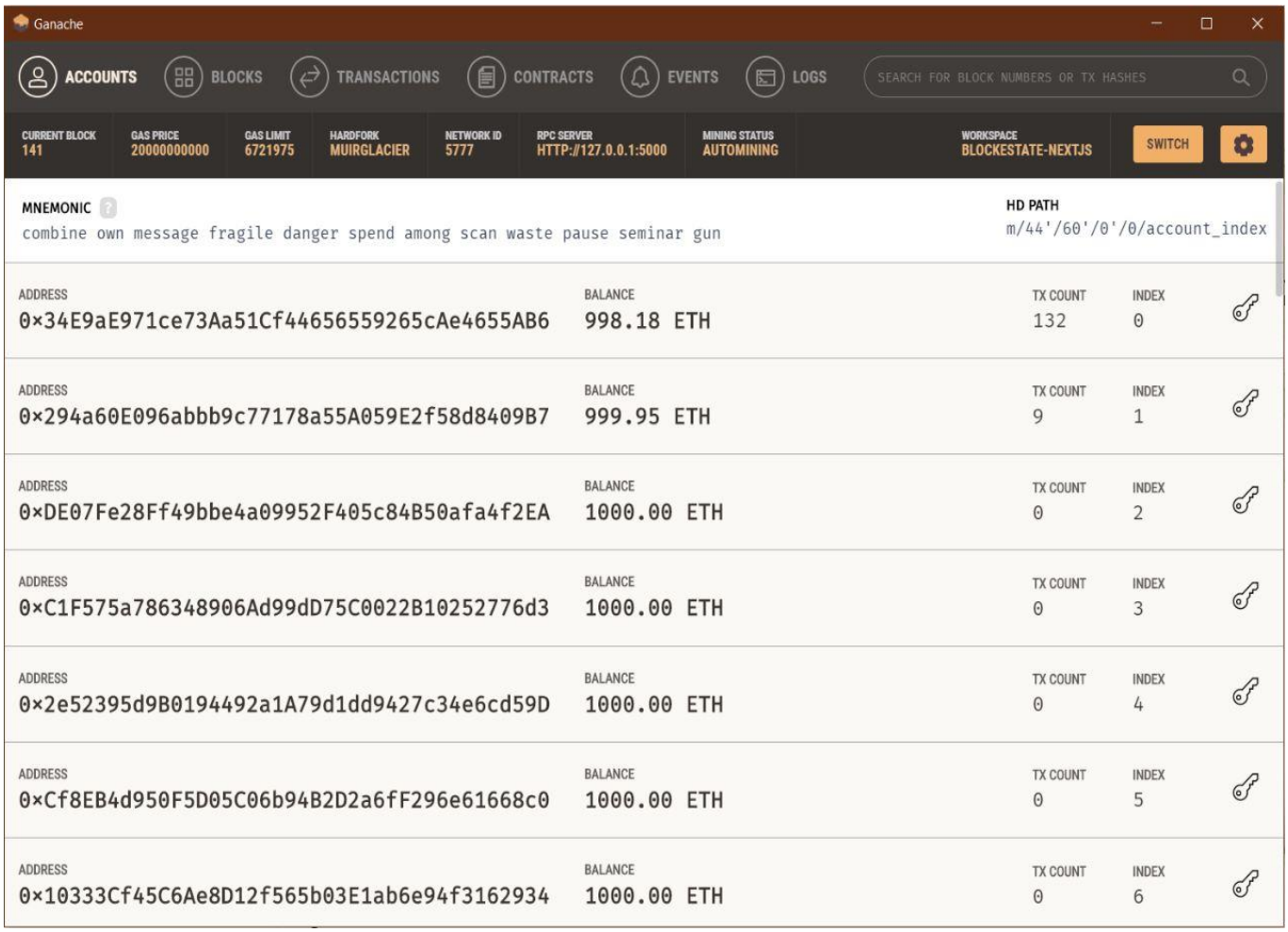


Figure 29: Ganache Workspace Setup

6. MetaMask Extension

We utilized the private key from Ganache and imported it into the MetaMask wallet to link the MetaMask extension to a replica of Ethereum wallet.

2.4. Project Components

1. Smart Contracts

A smart contract automatically upholds the terms and conditions that are spelt forth in it. It functions similarly to a computer program that runs on a secure network and may automate a variety of business processes, including payments, voting, and supply chain management. Smart contracts can lower costs and boost confidence since they are transparent and tamper-proof, maintained on a blockchain. They

allow for the automation of procedures without the necessity for a centralized authority, such as a bank or a government, and they can improve the effectiveness and fairness of interactions between people and organizations.

The Truffle configuration file (truffle-config.js), which we used to set up the Truffle framework and connect to a blockchain network, was immediately created when Truffle was installed.

In our instance, we used the Open Zeppelin library to create an ERC721 compatible smart contract for a land token. The Ethereum blockchain uses the ERC721 standard for non-fungible tokens (NFTs), which are singular tokens that cannot be swapped one for one for another token.

You can acquire a template for implementing the ERC721 standard and modify it to meet your needs by utilizing the Open Zeppelin library. For typical use cases, including maintaining ownership and metadata for NFTs, the library offers a selection of pre-built contracts and utilities.

a. Digital Land Token

Full Code for land Token is in appendix-D

The "Land Token" contract specifies a non-fungible token (NFT) that signifies ownership of a plot of land and shared owners function.

b. Compilation and Migration of Smart Contracts to the Blockchain

Run the following command to compile your contracts:

truffle compile

This generated the relevant JSON artefacts in the build/contracts/ directory and built all the contracts in the contracts/ directory.

To configure a migration, we created a new file in the migrations/directory:

migrations/1_initial_migration.js

migrations/2_deploy_tokens.js

Run the following command to migrate contracts to the blockchain.

truffle migrate

This will deploy all the migration in the migration/directory that are not deployed yet.

c. Middleware

A higher-order function called “requireAuth” is used to authenticate API calls. A handler function, which is the actual function that will be invoked after the request is authorized, is provided as an argument.

The 'apiKey' supplied in the request body is first compared to a pre-defined key inside the 'requireAuth' method. If so, the 'handler' function is used to process the request. A response of type 401 Unauthorized is returned if the key does not match.

By ensuring that only authorized users can access the protected endpoints, the 'requireAuth' method provides an additional layer of protection to the API.

The API key is securely hashed for authentication purposes using the bcryptjs package. Using bcryptjs.compareSync(), the user's API key is compared to the hashed API key kept on the server when a request is sent. As a result, the API key is always securely hashed before being saved or compared and is never communicated in plain text.

```
functionrequireAuth(handler) {
returnasync (req, res) => {

constisAuthenticated = bcryptjs.compareSync("APIs", req.body.apiKey)

if (isAuthenticated) {
returnawaithandler(req, res)
}
```

2. Login Page:

On this page, the Login API accepts a request and response object and determines whether the database contains any sessions with the supplied address and status "Active." A salt is created, and a string is hashed using the bycrypt. To secure user session data, a JSON Web Token (JWT) is utilized as a key with the hash that results. It returns the details of an already-existing session if one does.

A special session ID is created for the user's session using the uuidv4 library. The session will then be recognized in subsequent requests using this ID.

The tables in the postgreSQL database are being updated continuously throughout this procedure. The session table stores a user's session data after they log in. When a user registers, the user table is updated

with their personal data. When a new user signs in or registers, both tables are updated, otherwise only the session table is changed during login.

Appendix-C shows the code of above procedure.

3. Logout Page

If This function updates the status of a session in the database from “Active” to “Expired”

```
asyncfunctionhandler(req, res) {
const { address, session_id } = req.body

constlogout = awaitprisma.$queryRaw(Prisma.sql`Update sessions set status = 'Expired' where address =
${address.toLowerCase()} and session_id = ${session_id};`)//, [address, session_id]

res.json({
message:"User Logged Out"})}
```

4. Dashboard

a. Layout Page:

This React functional component establishes how a web application will look. It has features for logging in and out of the application using a MetaMask wallet and sending API queries to the backend, as well as rendering a header, sidebar, and main content area. sign-out link.

The 'load Contract' function loads an Application Binary Interface (ABI) for a contract from a JSON file and uses the 'truffle-contract' library to generate an instance of the contract. The provider (Web3 provider object) and the contract's name are its two required arguments. Using the "fetch()" function, it first retrieves the JSON file from the "/contracts" folder before using the "contract(Artifact)" function from the "truffle-contract" library to construct a contract object from the Artefact. Finally, using '_contract. Deployed()' to return the deployed contract object after setting the contract's provider.

```
importcontractfrom"@truffle/contract"
exportconstloadContract=async (name, provider) => {
constres=awaitfetch(`/contracts/${name}.json`)
constArtifact=awaitres.json()

const_contract=contract(Artifact)
```

```
_contract.setProvider(provider)

constdeployedContract=await_contract.deployed()

returndeployedContract
}
```

b. Web Map:

The Leaflet package is used to render a map by the React component called Map Component. It uses an OpenStreetMap Tile Layer as the foundation layer and shows a map focused on a particular spot. To depict the user's and other users' owned lands on the map, the component draws polygons. Each polygon has a distinct color and a message-containing Popup.

Please refer to Appendix-C for code.

5. My properties page:

This functional React component renders a list of attributes that a user owns. A Web3 API object, user information, an API key, and the user's Ethereum wallet address are given to it as props.

Please refer to Appendix-C for code.

6. My Account Page:

This React component creates a form that users may utilize to update their account information. Using an API endpoint called "UpdateUserDetails," the component retrieves the user's most recent information from the backend and updates it when the form is submitted.

A list of things is displayed using this React component. It refreshes the state after retrieving the objects from an API endpoint. The "LandSaleCard" component is then used by the component to render the list of items.

7. Marketplace:

```
constMarketplaceComp = ({ address, web3Api, apiKey }) => {
```

```
const [landSale, setLandSale] = useState([])

useEffect(() => {
constgetAllLandSale = async () => {

if(!address) return

const { landToken, web3 } = web3Api

constresponse = awaitfetch("api/landSale/getAllLandSale", {
method:"POST",
body:JSON.stringify({ address, apiKey }),
headers: {
'Content-Type':'application/json'
}
})

constdata = awaitresponse.json()

if(!data) return

if(data.landOnSale) {
constupdatedLandOnSale = data.landOnSale.map(asyncland=> {

consttokenURI = awaitlandToken.tokenURI(parseInt(land.land_id))

return {
...land,
geometry:JSON.parse(tokenURI).geometry
}
})
constresults = awaitPromise.all(updatedLandOnSale)
setLandSale(results)
}
}
web3Api.web3&&address&&getAllLandSale()
}, [web3Api.web3&&address])
```

8. Ongoing Transaction Page

This React component illustrates active transactions. Address, router, and API Key are the three props that are accepted. The component refreshes the state with the retrieved information after retrieving a list of active transactions using an HTTP POST call to an API endpoint. Each transaction is then represented as a card with the address of the buyer or seller, the approved price, and a button to complete the transaction.

```
constOngoingComp = ({ address, router, apiKey }) => {
const [transactionBid, setTransactionBid] = useState([])
useEffect(() => {
constgetOngoingTransaction = async () => {
constresponse = awaitfetch("api/transaction/getTransactionBidList", {
method:"POST",
body:JSON.stringify({ address, apiKey }),
headers: {
'Content-Type':'application/json'
}
})
constdata = awaitresponse.json()
if(!data) return
setTransactionBid(data.transactionBid)
}
address&&getOngoingTransaction()
}, [address])
```

9. Verify Land Page

The provided React component generates a token for a land using a Web3 API and uses an API to confirm the presence of a land. It controls the component's state and shows the user messages. The component has state-based conditional rendering for the form and messages.

Please refer to Appendix-C for code.

10. Transaction History Page

This React functional component shows the buying and selling history of the lands owned by a certain user. The 'land Token' contract instance used to retrieve the transaction history using the 'getPastEvents' method is contained in the 'web3Api' object. Using the 'retrieveTransferDocument' method, the fetched token IDs are then utilized to find the matching land transfer document. The documents are downloaded, parsed into JSON, and then saved in the 'soldHistoryLand' and 'boughtHistoryLand' states, respectively. After then, the component presents the return statement's transaction history.

a. Transfer of Land:

Before the transfer may happen, both the buyer and the seller must certify that the deal has been closed and the money has cleared. They can push a button to start the land transfer when both sides have confirmed.

It allows the user to sign and approve the transaction while retrieving user and transaction information from API endpoints. Additionally, the component shows details about the transaction and the user's participation in it. The element is made to be a component of a bigger web application.

Please refer to Appendix-C for code.

2.5. Deployment

1. AWS/RDS:

• Set up an account with AWS.

• Created a PostgreSQL database using AWS's RDS (Relational Database Service).

• To specify the layout of database tables, a PostgreSQL schema was uploaded to the RDS instance.

2. Database Migration with Prisma:

• In addition to the AWS/RDS configuration, Prisma was used to communicate with your PostgreSQL database.

• Created migration files for the AWS/RDS PostgreSQL database based on database schema updates using Prisma's CLI commands.

3. MetaMask and Sepolia:

• MetaMask was installed and linked to the Sepolia network. Projects can communicate with the blockchain network using this setup.

4. Network Configuration:

The project was set up to connect to the specified blockchain network using MetaMask and Sepolia. This network configuration allows the project to communicate with the blockchain, conduct transactions, and obtain blockchain-related data.

Frontend Deployment using Vercel:

- Used Vercel, a website hosting platform, for the front-end deployment.
- By linking the project's repository to Vercel, it is now able to automatically generate and distribute front-end code.

- Vercel housed front-end code and offered a public URL where users could access your deployed application. • Once linked, Vercel's continuous deployment capability triggered deployments anytime changes were submitted to the repository.
- We were able to launch our project successfully by utilizing Vercel for frontend hosting and deployment and AWS for backend infrastructure (RDS and database migration with Prisma). The application is currently accessible to users via the deployed frontend hosted on Vercel.

RESULTS AND DISCUSSION

3.1. Results

This section provides outcome of the web application created. The pages of our web application are shown in form of screenshots and their description is provided. The marketplace page for browsing and bidding on properties, a verification page to confirm the legality of properties, an ongoing transaction page to follow the status of deals, and a My Account page to manage personal information are just a few of the essential features included in this DApp.

Login Page

The "Login page" of the DApp gives users the option to connect after which the MetaMask extension appears. If user is logged into their MetaMask account, the system will automatically fetch their wallet address; if not, it will first prompt them to log in before fetching their address to log into the app. The security of the user's personal data and financial transactions is ensured by the direct connection of these options to the Ethereum wallet. First-time users must enter their CNIC, then other personal information including their complete name, address, and phone number in order to register on the DApp. To ensure that each user is distinct and to stop any fraudulent behavior, this information is required. Figure 34 to Figure 54 show complete app interfaces and transfer process.

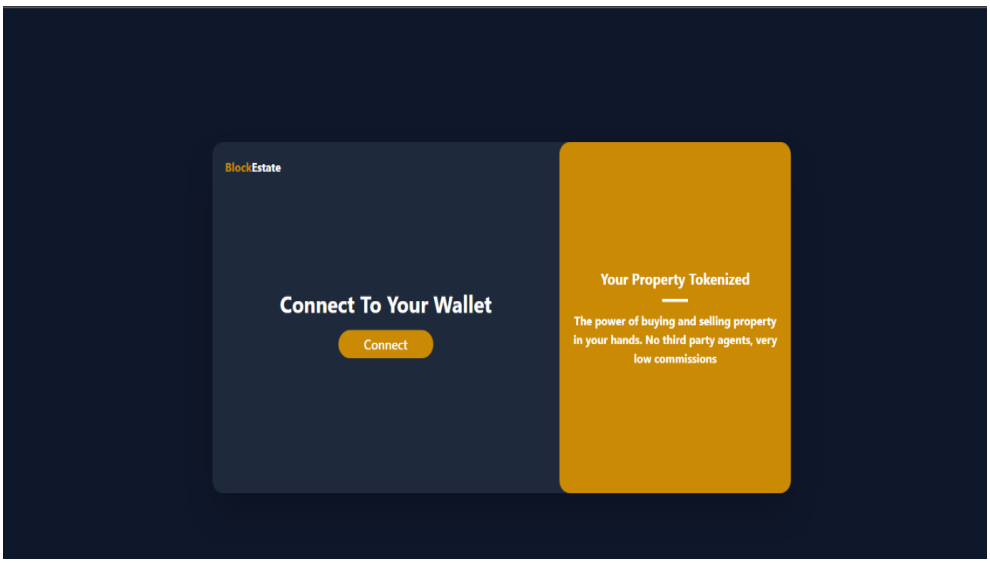


Figure 30: Connect to Wallet

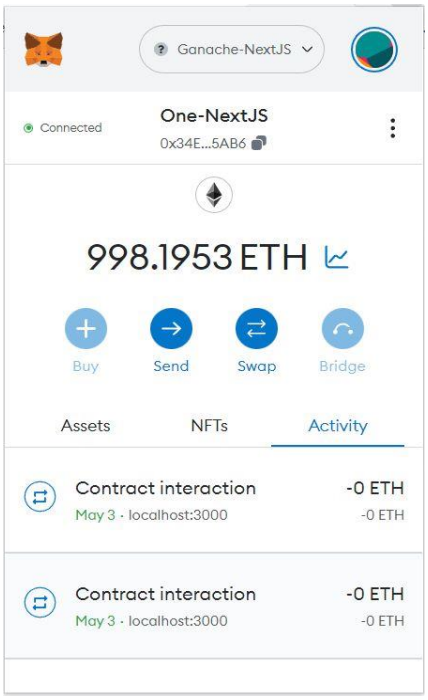


Figure 31: MetaMask Extension

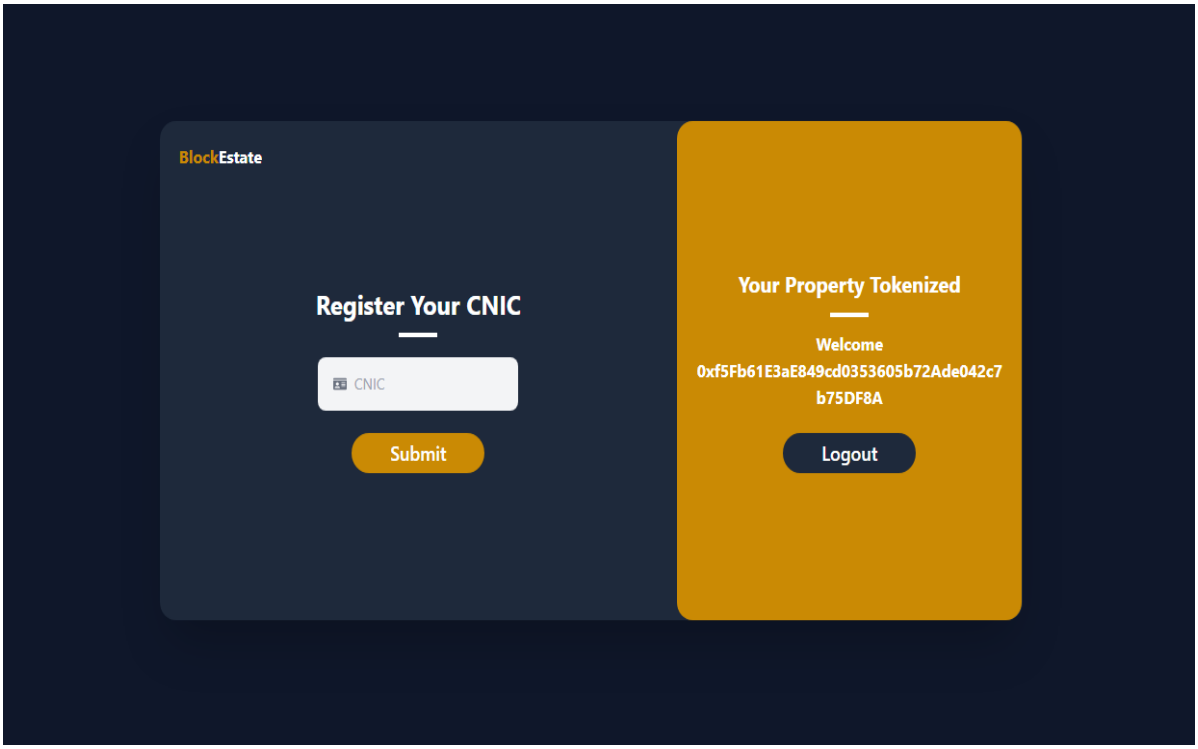


Figure 32: Register your CNIC

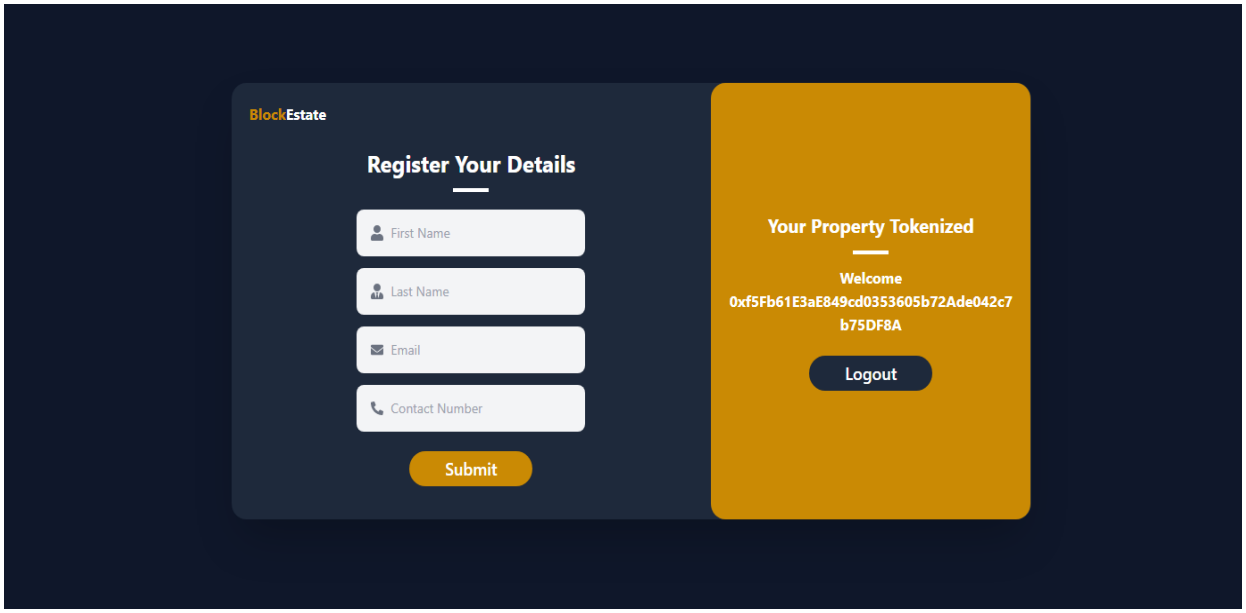


Figure 33: Register your Details

The registration process will be unsuccessful if the same user tries to register again using the same information. This technology guarantees the individuality of each user on the platform and thwarts any potential fraud.

Dashboard

The "Dashboard" of the DApp is a user-friendly interface that offers users navigation options and shows basic DApp's information. The dashboard's web map shows the user's properties as well as the properties of other platform users that are on sale. You can look through the properties on this map to locate prospective investment opportunities. Users can zoom in and out of the interactive map. Buyers who are keen in making an investment in a particular neighborhood or kind of property may find this option to be especially helpful.

Users get quick access to the platform's various pages via the sidebar on the dashboard. Users can access and change their personal data, including their name, address, and contact details, on the "My Account" page. The "Home Page" returns users to the primary dashboard. All the user's properties, together with their specifics and status, are shown on the "My Properties" page. Users can confirm their identity on the "Verification Page" to establish more credibility on the network. The details of any ongoing

transactions, including their status and progression, are displayed on the "Ongoing Transaction" page. They can view the properties on sale on “Marketplace” page. The details of all the previous transaction are on “Transaction History” page.

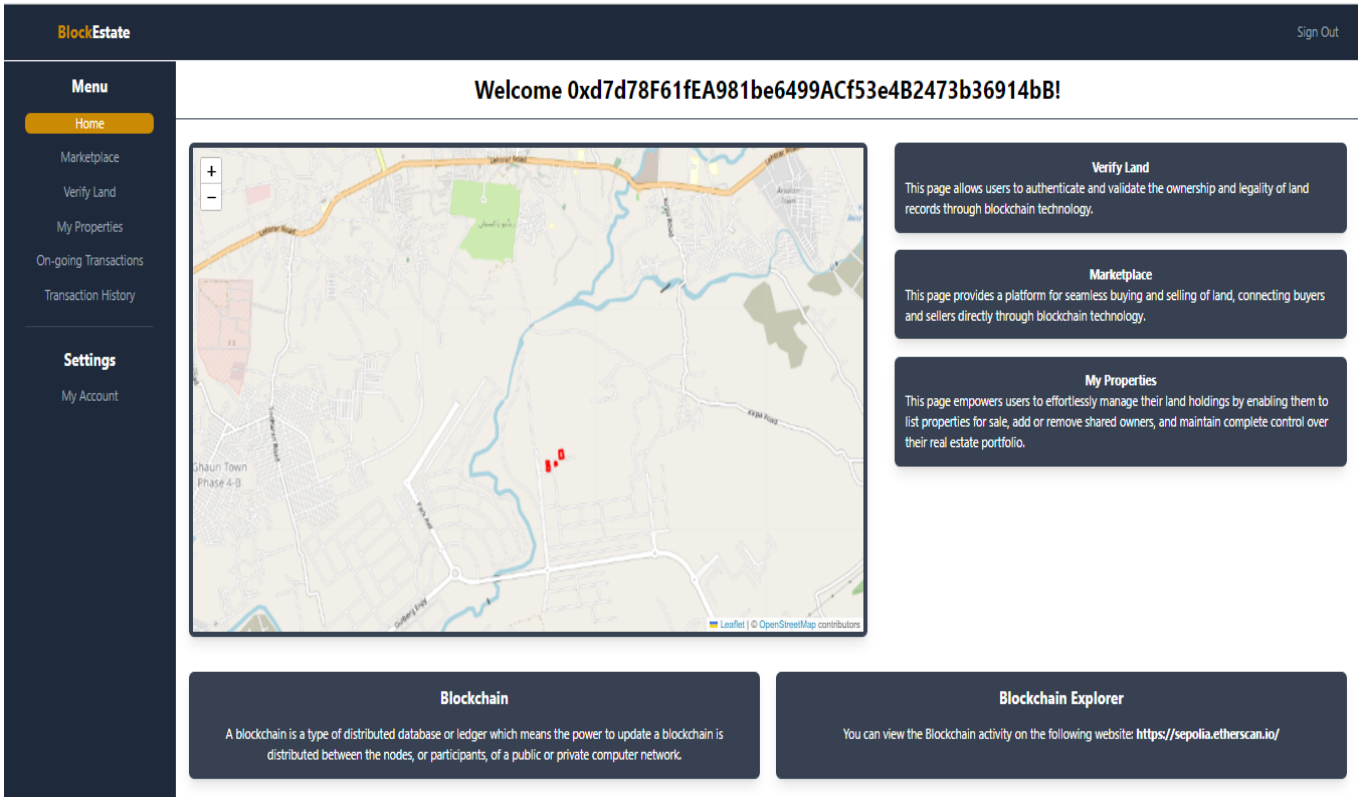


Figure 34: Dashboard

My Properties

Users can manage their owned lands/properties within the DApp by visiting the "My Properties" tab. In addition, a web map displaying the location of the user's properties is shown in the cards. A distinct plot of land is displayed on each card.

The ability for the user to manage their properties is the major goal of this page. Setting the option to sell the property, determining the asking price for the property, and accepting or rejecting a bid request on the property are all included in this. It also involves changing the number of shared owners.

Multiple users may have an interest in the same property thanks to the DApp's shared ownership functionality for properties. The user can select the "Add Owners" option for the desired property to set shared ownership and then add shared owners as necessary. For families or groups of people who want to share ownership of a property, this function is especially helpful.

The user can set the status for sale, they can put the property for sale or they can remove it from sale. As a result, interested parties can inspect the property and submit bids. The bid request can then be viewed and accepted or rejected by them by going to the "View Bid Request" tab. The smart contract transfers ownership of the property to the buyer after going through whole transaction phase.

Appendix-A provide complete procedure of working of marketplace.

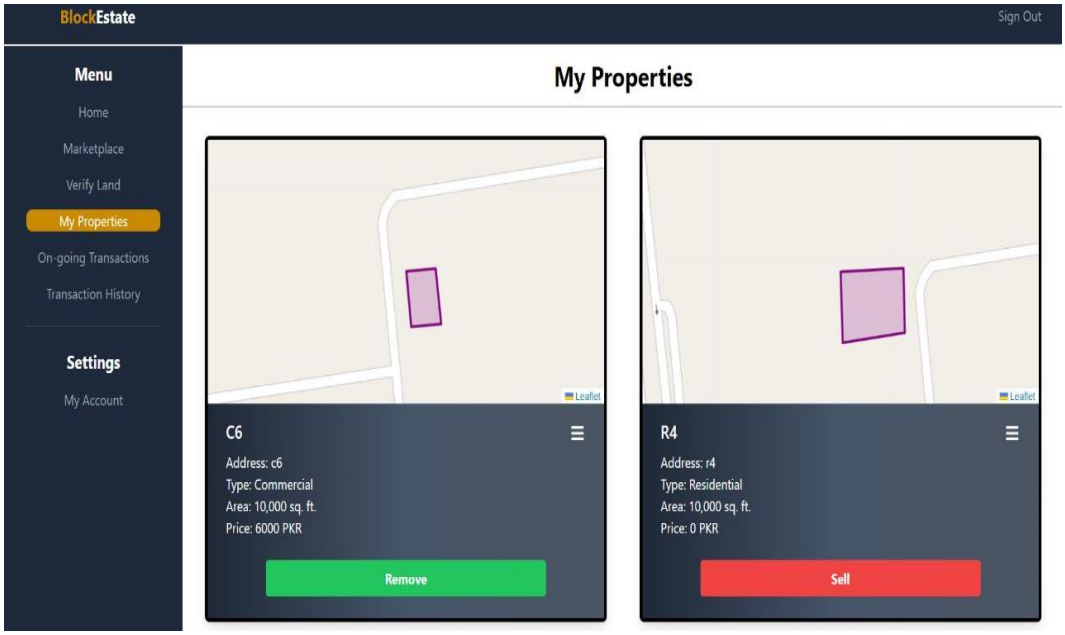


Figure 35: My Properties Page

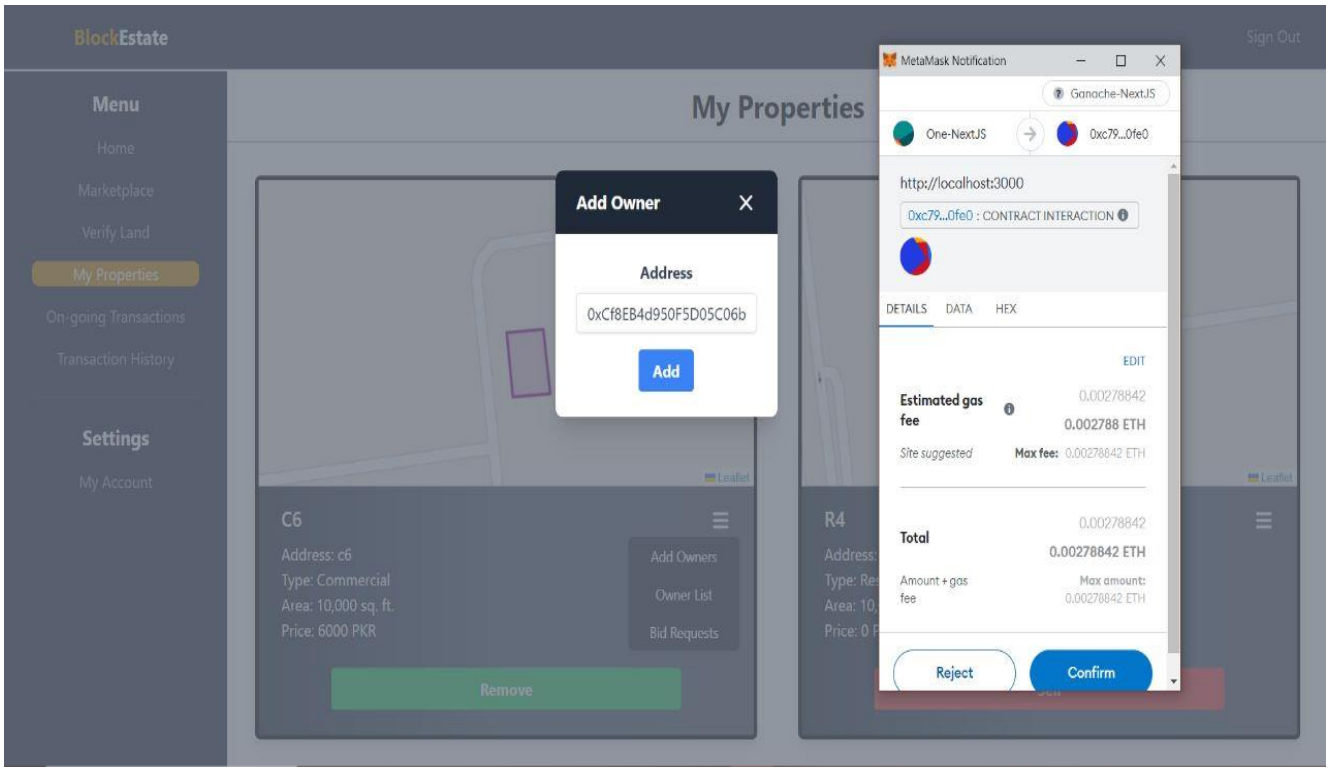


Figure 36: Add Shared Owners

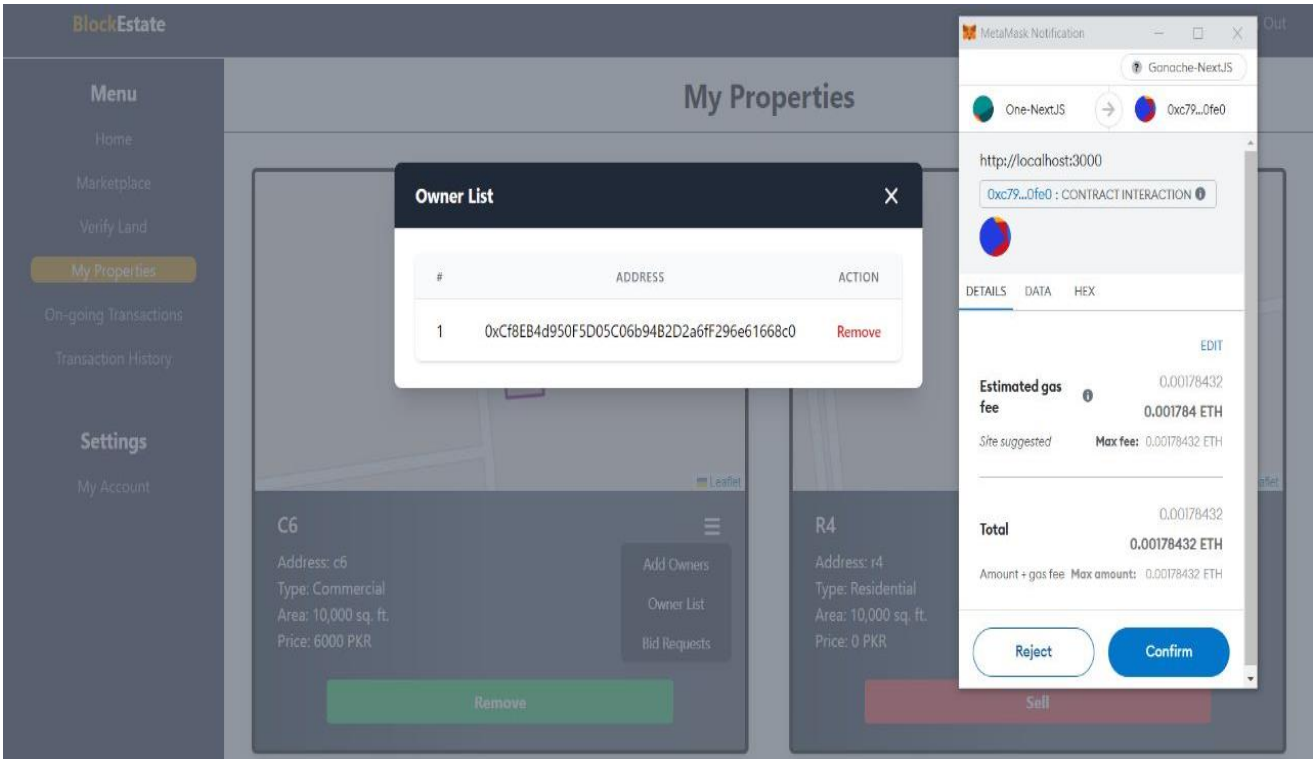


Figure 37: Add Shared Owners

Marketplace

The "Marketplace" feature, which enables users to browse through all the properties that are up for sale on the platform, is a crucial part of the DApp. All of the active properties that are up for sale right now are shown on the website by their respective owners. The location, size, and asking price of each property are included in a tile format that is used to display them.

Users may more easily visualize each property's surroundings and adjacent amenities thanks to the online map that is displayed on the page. Users may rapidly identify the properties that match their interests.

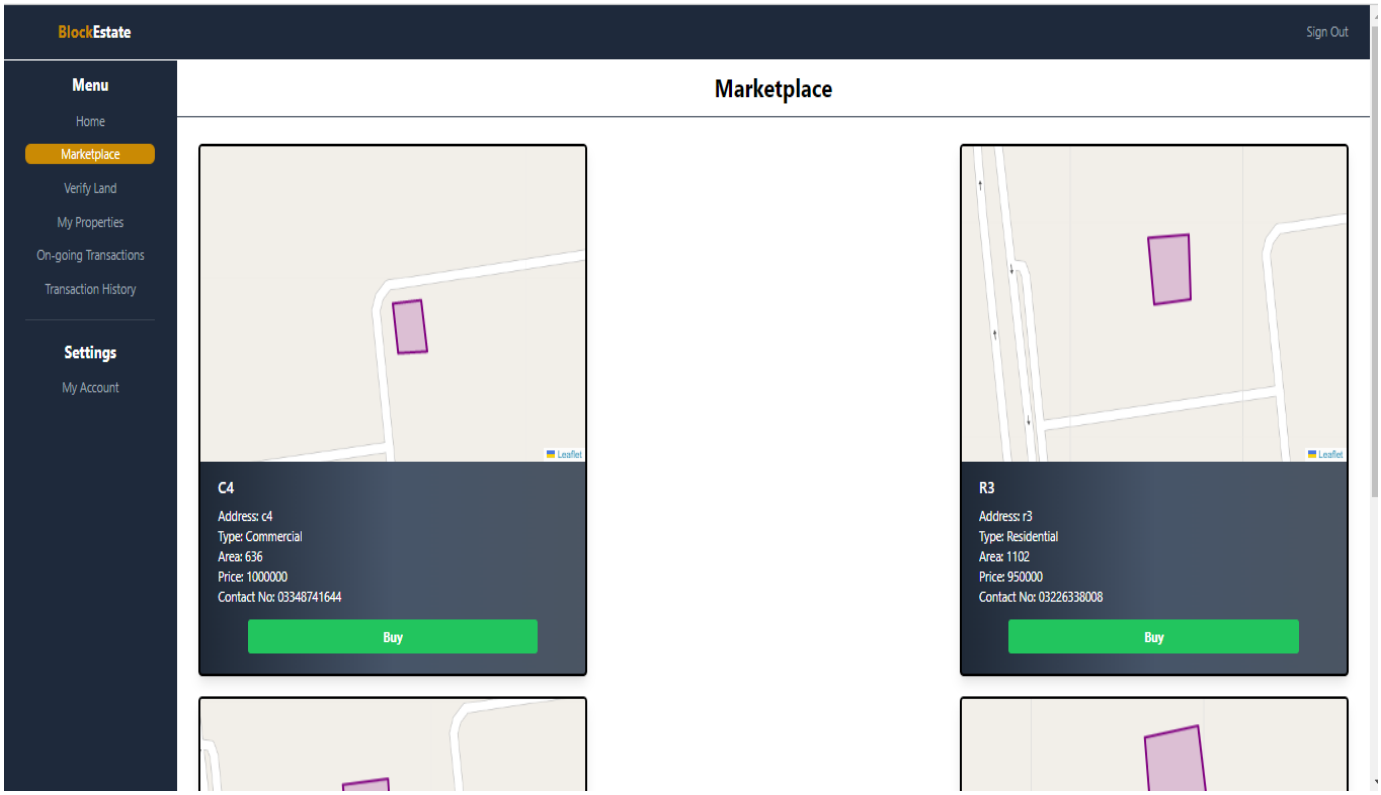


Figure 38: Marketplace

Making an offer on a property is one of the main elements of the Marketplace page. The "Buy" button will take the user to the page where they can place a bid. The page allows for the entry of the bid amount, and after submission, the bid is delivered to the property owner for evaluation.

The offer is then up for acceptance or rejection by the seller. If the offer is accepted, the transfer process of smart contract will automatically run after the payment has cleared, giving the buyer ownership of the property. If the offer is turned down, the house will still be up for sale, and potential purchasers can place further bids until it sells.

Buyer

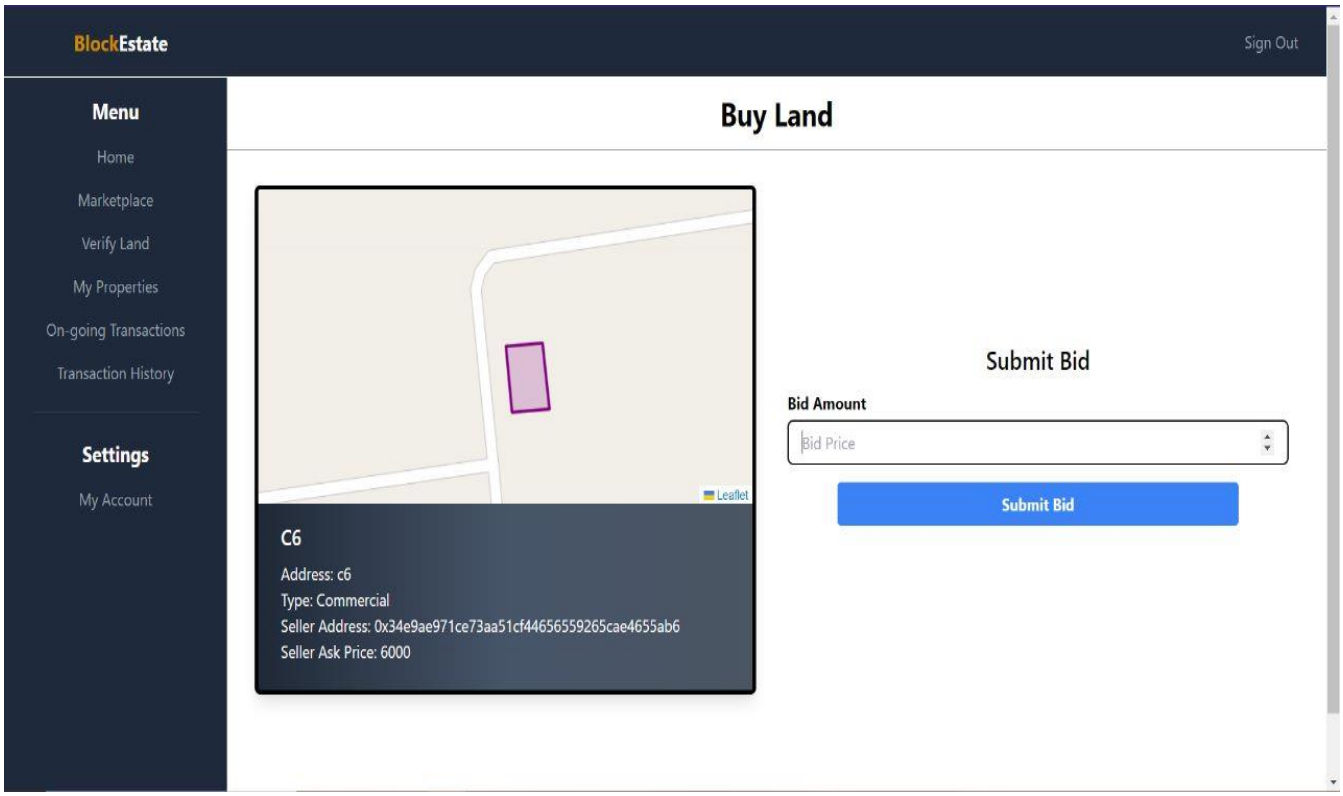


Figure 39: Submit Bid Page

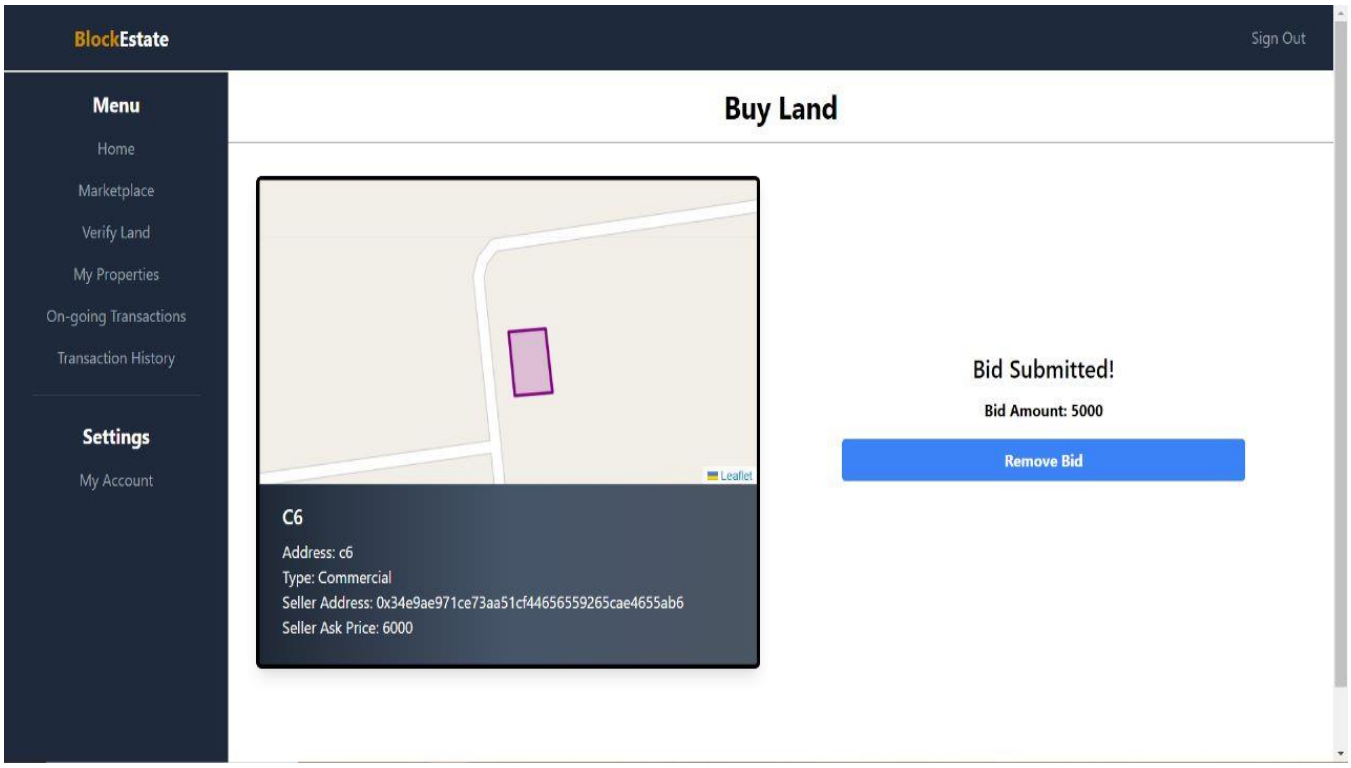


Figure 40: Bid Submitted Page

Seller

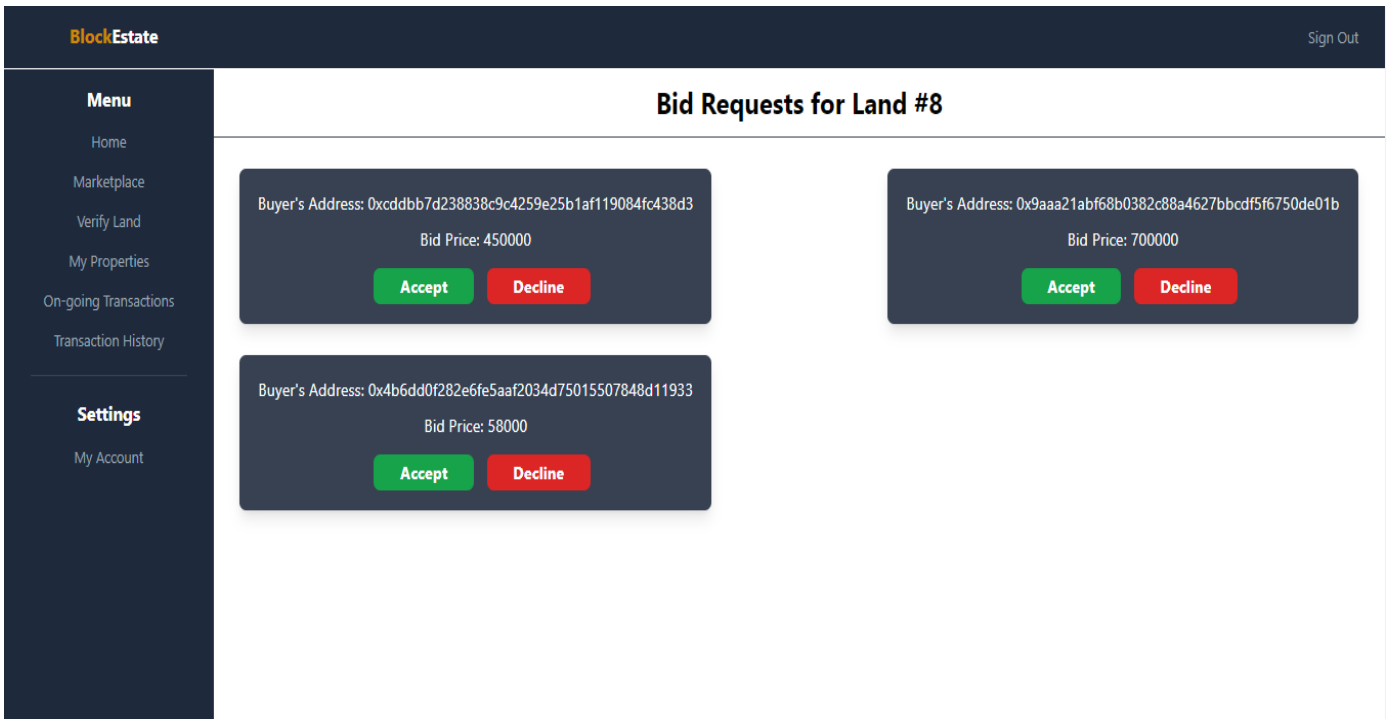


Figure 41: Bid Requests Page

Verification

Our decentralized application includes a verification page the user can verify their lands. The "Verification" page is meant to be straightforward and simple to use. The land ID for the property and user' CNIC must be entered must be entered when they access this page, the land id will entered manually whereas CNIC will be fetched by the system. The system will then examine the database to see if the land associated with this ID already exists. The user will be given the option to create the digital token of the land if the land is confirmed to exist.

The verification process requires that a digital token be created, which is a crucial step. It makes it practically hard for anyone to tamper with or change the record by creating a distinct, immutable record of the land on the blockchain. The digital token acts as evidence of ownership transfer.

Features to stop fraudulent activity and duplicate registrations are present on the verification page. For instance, the system will catch it and stop someone from registering the same piece of land twice with different IDs. The system will also deny the registration if the landowner tries to register a plot of land that is already registered to someone else.

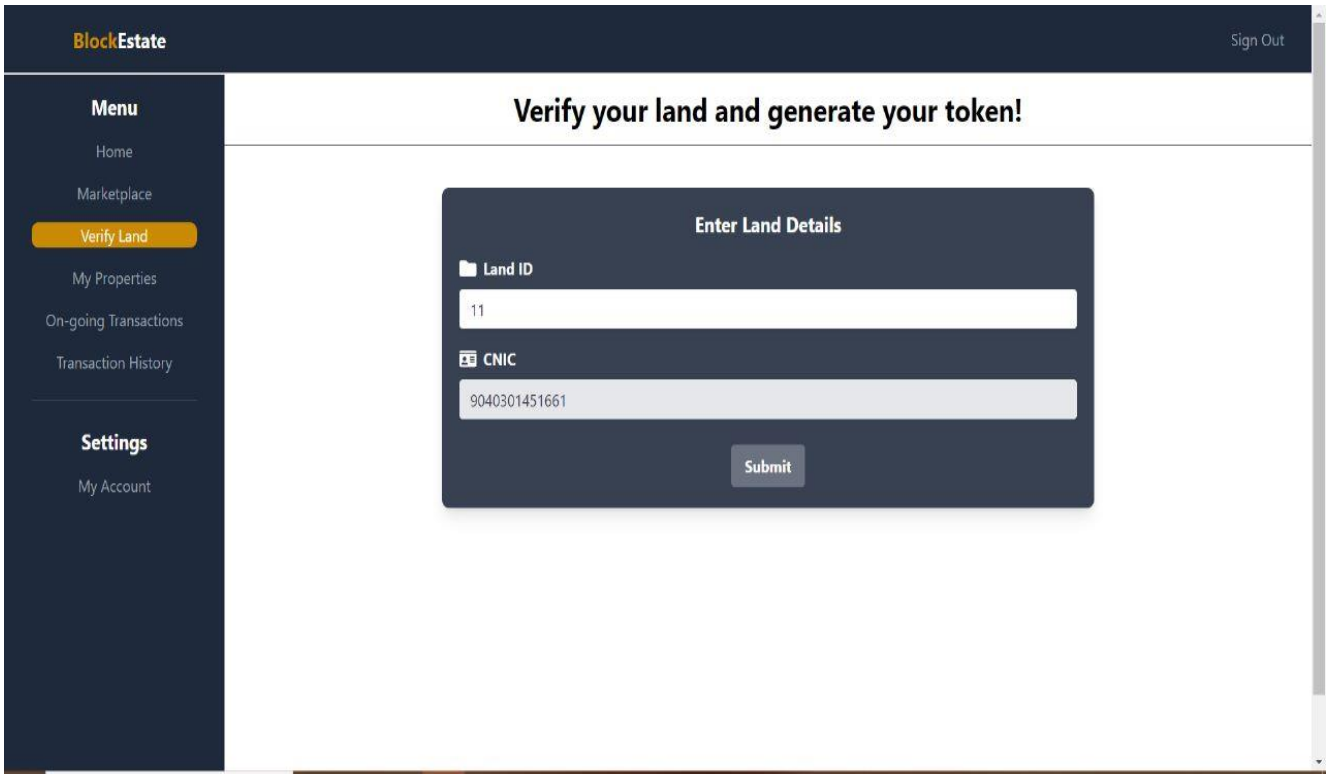


Figure 42: Land Verification

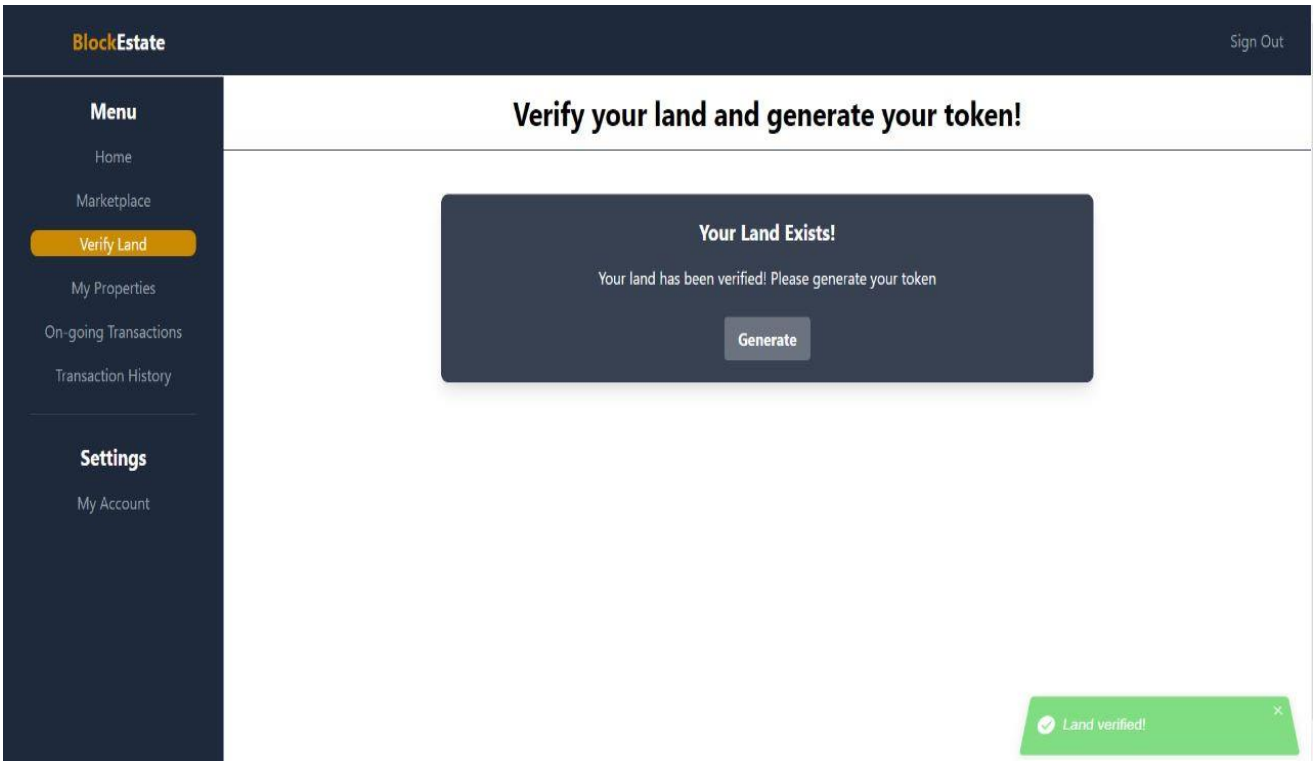


Figure 43: Token Generation (A)

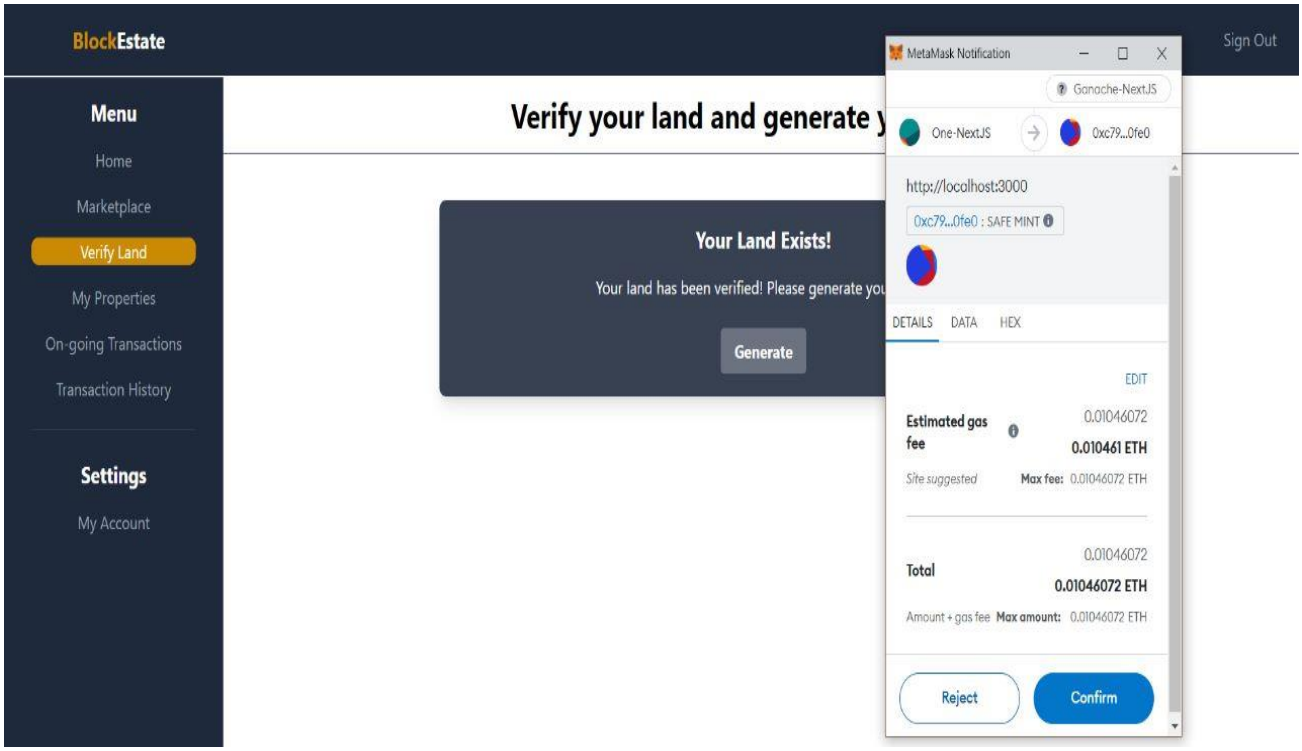


Figure 44: Token Generation (B)

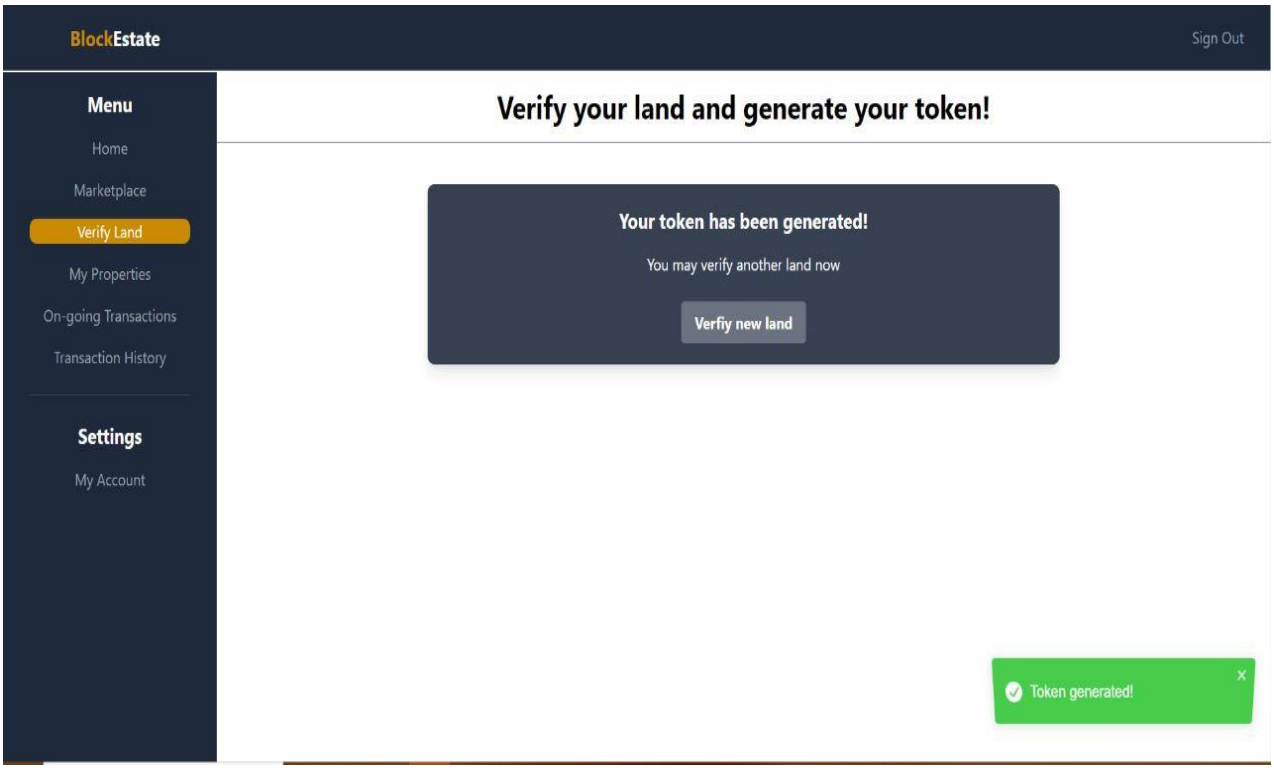


Figure 45: Token Generated

Appendix-A shows the procedure and libraries

Ongoing Transactions

Users can obtain a detailed view of any ongoing transactions, including their status and progress, on the ongoing transaction page of app. Users who have placed a bid on a property and wish to follow the development of the deal will find this website to be of particular use.

The DApp automatically changes the ongoing transaction page to reflect the new transaction when a user submits a bid on a property and that bid is approved.

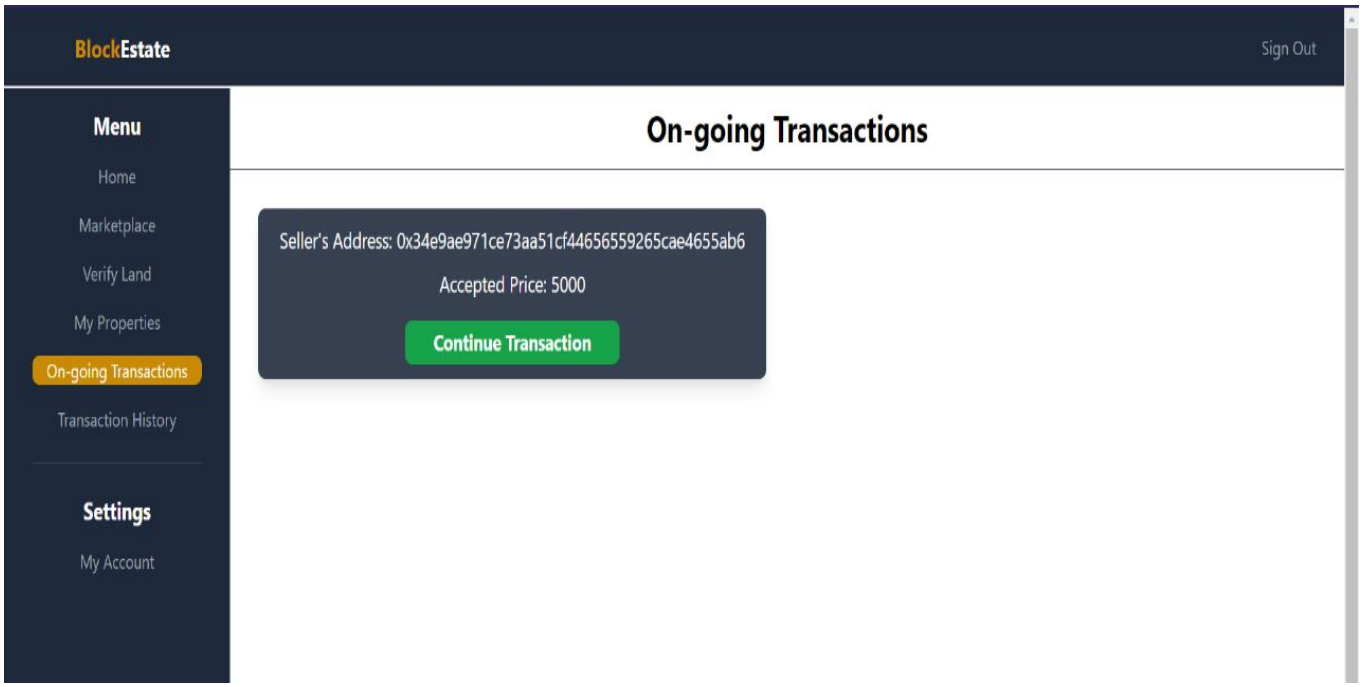


Figure 46: Ongoing Transactions Page

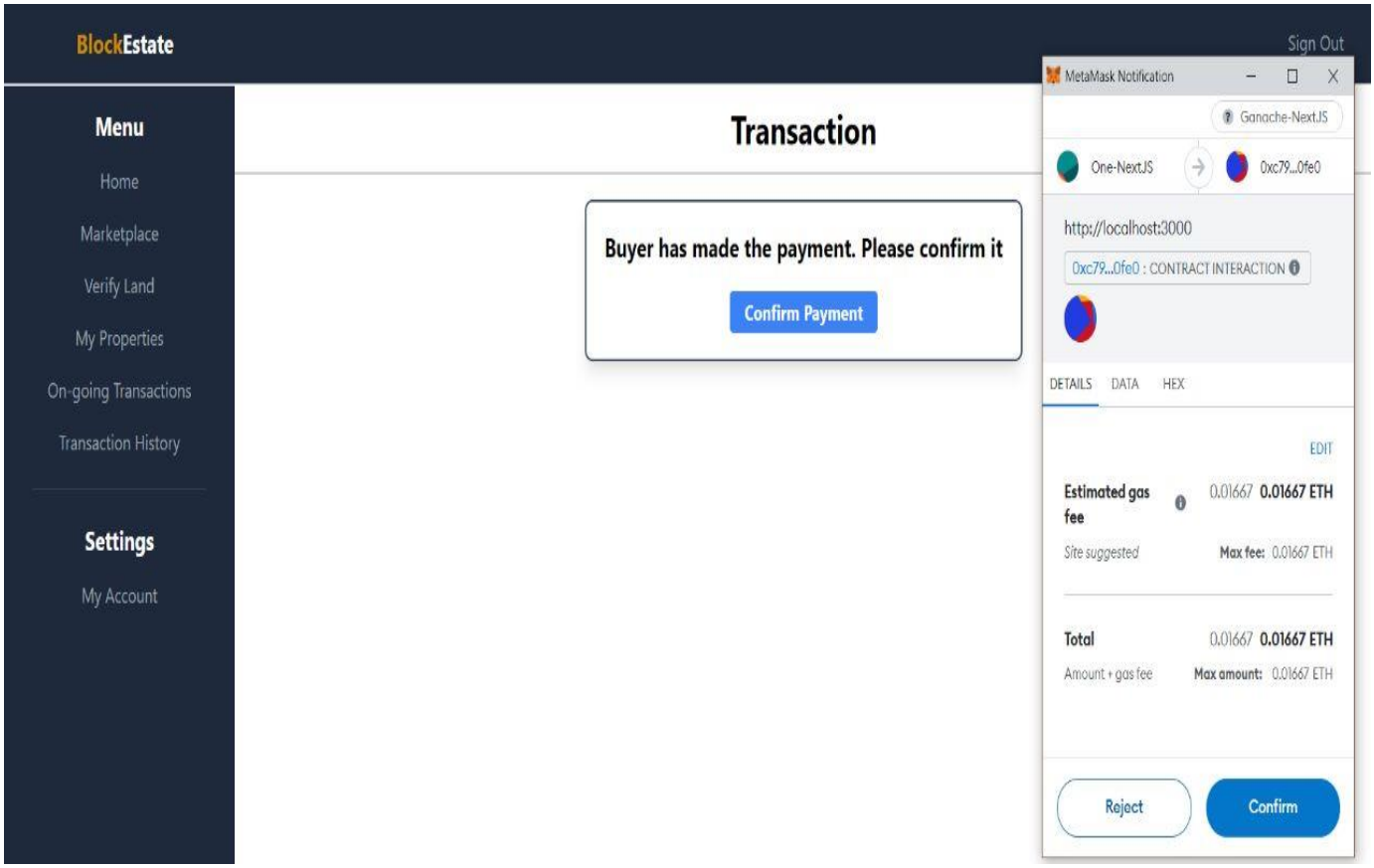


Figure 47: Payment Confirmation

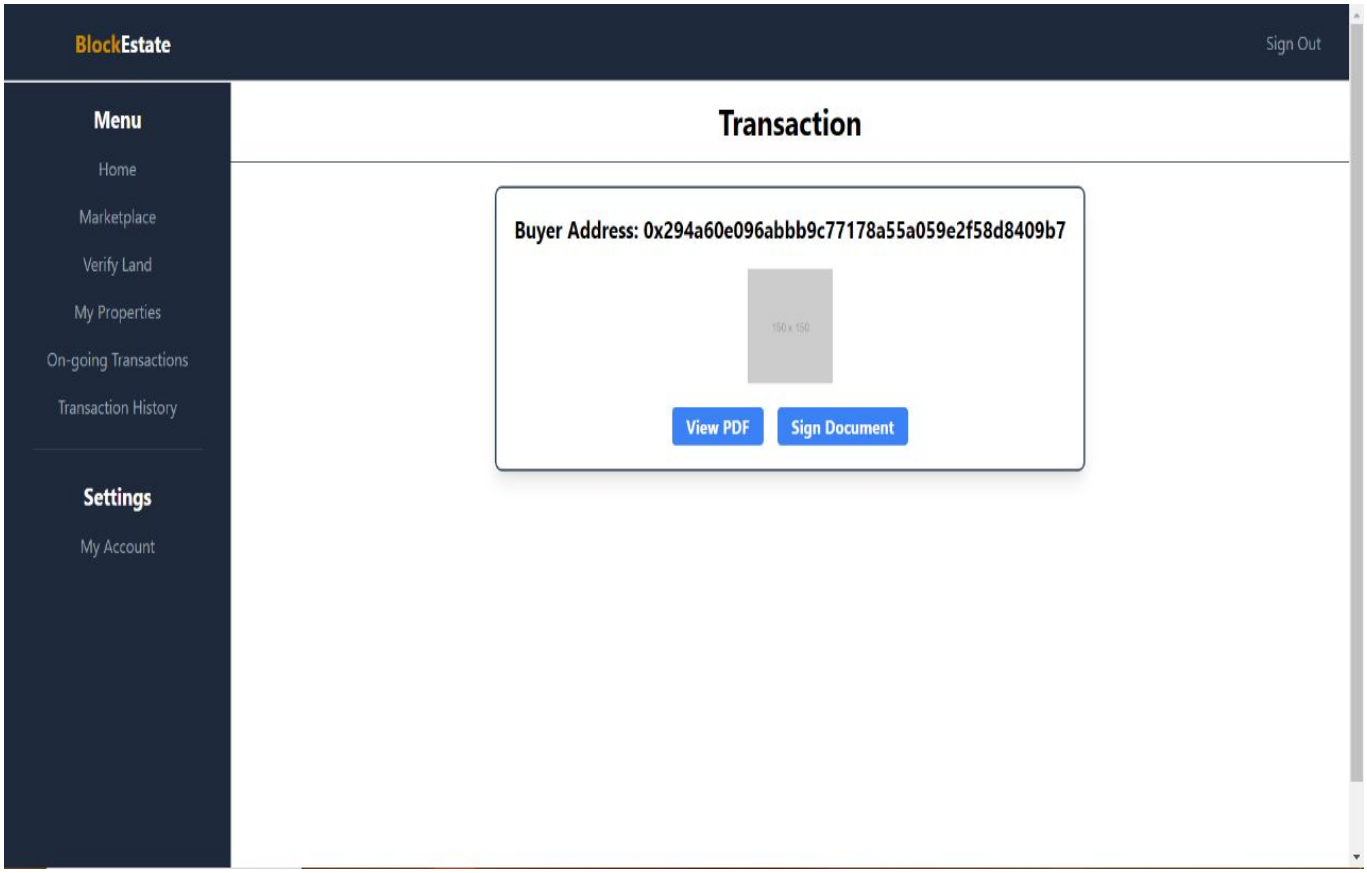


Figure 48: Sign Contract

My Account

The user can modify their personal data, including their complete name, email, and phone number, on the "My Account" tab of our DApp. Users will have complete control over their account information through this page, hence increasing accessibility.

The page is created in a way that makes it simple for the user to change their personal information without becoming lost. The user may see their wallet address and CNIC at the top of the page. The user cannot modify their CNIC or wallet address, which is a significant element of the My Account page. The purpose of this limitation is to prevent users from opening multiple accounts using the same CNIC or wallet address and to verify that their identity is validated. This limitation increases the security of our DApp and helps to thwart fraudulent activity.

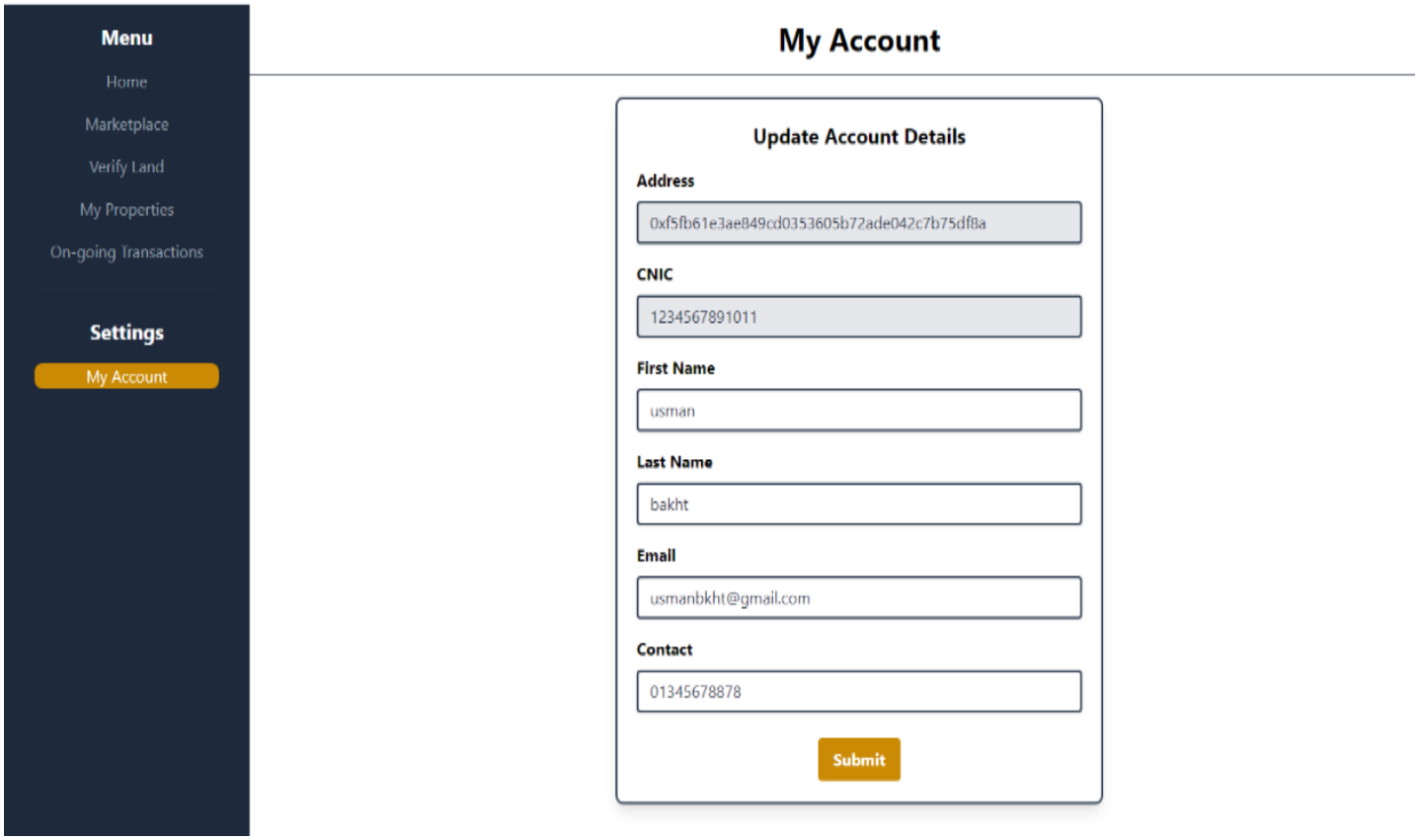


Figure 49: Update Account Details

Transaction History

Users can simply access their previous transaction records on the "Transaction History" page, giving them a thorough overview of their transaction history.

Users can access and examine their Legal Agreements on this page in addition to transaction history. These agreements give consumers a point of reference and written proof of their agreements by containing the legally enforceable terms and conditions related to their property transactions.

It contains all the transaction records for lands that they have previously bought/sold. User also have an option to download a pdf agreement as a legal agreement in pdf form, which contains all details of buyer and seller

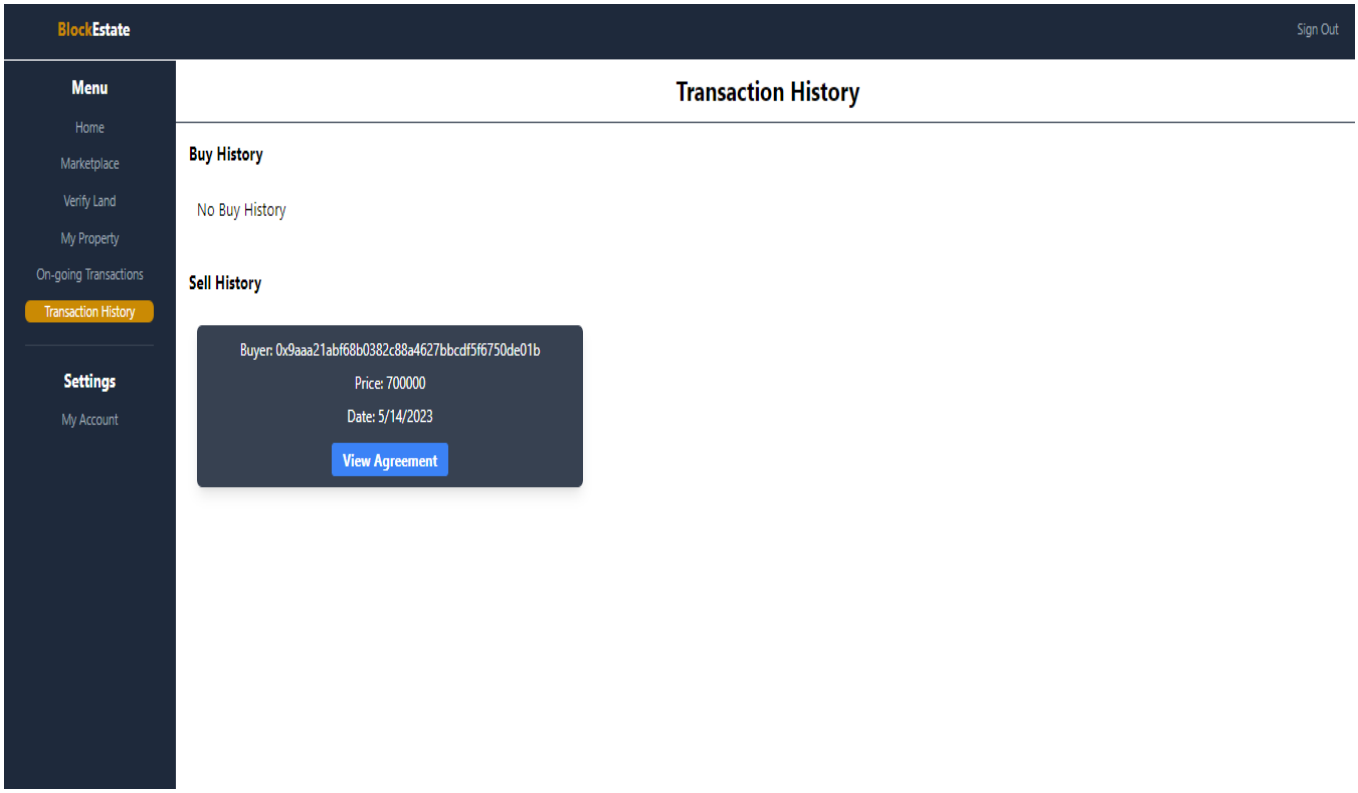


Figure 50: Transaction History Page

Testing

4. Our deployed application was tested by 4 people. The individuals divided up the sample land shapefiles among themselves. Each of the four individuals owned a single plot of property. Now, during the initial test run, we encountered a few problems and bugs that were fixed before the second and final test run.
5. Each user was now able to log in to their separate accounts following registration of the users using their CNIC and database verification. After that, they clicked the "My Property" button where they could access their own land property, sell it, add co-owners, etc.
6. Now that Person A has verified his Land "R1" for sale, it is up for bids from the other three. Then Person A selected the most favorable offer and continued with the transfer procedure. Refer to the bidder whose offer was approved as B, both of them now proceed to the transferring process. The following step is for both parties to sign the legal agreement of sale and then confirm the payment by uploading their respective payment receipts. Each will upload and wait for the verification process to match it. Now that the land transfer had been confirmed, on screen of Person A,

MetaMask extension will pop up. A gas fee would be deducted from his wallet when he approves it, ensuring the transfer of land title. Finally, they may see the signed contract, and Person B is now the owner of the land. The "Transaction History" button also displays current information. Similar to that, four people worked together to identify issues that needed additional correction.

3.2. Discussion

Being a Geoinformatics Engineer, we have attempted to integrate latest tools and technologies available to the date. Integration of Spatial data with blockchain and Making a Decentralized Web application, which supports buying and selling of land digitally, required a lot of learning, in both technical and non-technical way. We had to work hard in order to find technological gap, doing surveys, learning about blockchain and integrating technologies. It has been a huge achievement for us. We have used open source technologies in order to increase the flexibility and interoperability of our web app. From using basics of web to learning Frameworks like Next.js, Prisma for PostgreSQL, and web 3.0 technologies like blockchain, Solidity, Smart Contracts, we have gained lot of experience. Current land ownership process is tedious, transactions are slow, inefficient, and costly, and a decentralized approach can improve transparency, reduce fees, and empower individuals to conduct land transactions more easily. We have learnt about concepts, structure and working of blockchain. We have learned about Blocks, Hash, minting, and linking GeoJSON with smart contracts. Throughout the project, we went through different stages, which include:

- Research about current land ownership records and transfer process.
- Information about existing solutions
- Open-source tools and technologies required.
- Choosing the flexible and scalable technology stack.
- Installing Frameworks, Designing front-end UI pages.
- Develop the smart contract(s) for the app using Solidity or another programming language. Test the contracts using a local blockchain emulator Ganache.
- Connecting Backend with frontend using next.js

With successful running application, there were some limitations and bottleneck situations we had to face; some of them are listed as followed:

- Sending the data on blockchain efficiently was challenging task, as to minimize the gas fee by changing data on blockchain less frequently.
- Limitations on data formats was a limitation as JavaScript and Leaflet cannot process shapefile (.shp) directly. They were manually converted to GeoJSON. A python script was used for this purpose.

- In order to make our web application work across different platforms, we have to make it responsive and we have to design UI accordingly
- There were some constraints while deploying the app on web as there was no single platform which can host all frontend, backend, smart contracts and Databases.

These limitations are to be considered for their solution. For now, they can be solved one way or another.

CONCLUSION AND RECOMMENDATIONS

4.1. Conclusion

Blockchain and GIS made us adopt a new approach and create a decentralized real estate buying and selling application. Our application is provides a streamlined and hassle free approach towards the whole process. The interface of the application allows individuals to register as well as upload their land records with ease, requiring a very minimal gas fee. The land, can either be put up for sale, shared owners can be added or removed with a high level of security. The application assures a smooth land transfer. Our solution ensures that transactions are secure and tamper-proof, while the integration of GIS mapping technology allows for more accurate and comprehensive representation of the land parcel being bought or sold. In conclusion, our decentralized application represents a significant step forward in the realm of land transactions, providing a more efficient and secure method for peer to peer buying and selling of land.

4.2. Recommendations

Our decentralized application seeks to provide a readily available platform for users to buy and sell land in a streamlined and secure manner. However, for such a system to be truly effective, one must implement it on a larger, national scale. This would enable the verification of registered land and its owners, which is crucial for the success of the project. To achieve this, a connection with the databases of land authorities and NADRA is essential.

By providing the public with an app that operates without the need for a central authority, we can offer a more efficient and cost-effective solution compared to traditional methods that require the involvement of a real estate agent. This approach not only benefits the state but also the citizens by offering a scam-free environment that ensures a fair and secure transaction process. Ultimately, our goal is to create a solution that benefits all stakeholders involved in the land-buying and selling process, providing them with a transparent, efficient, and cost-effective way to conduct transactions.

REFERENCES

1. Ahsan, M., Hussain, E., & Ali, Z. (2017). Integrated Geospatial Evaluation of Manual Cadastral Mapping: a case study of Pakistan. *Survey Review*, *49*(356), 355–369. https://doi.org/10.1080/00396265.2016.1180755
2. Karamitsos, I., Mavroudis, D., & Barghuthi, N. B. A. (2018). Design of the Blockchain Smart Contract: A Use Case for Real Estate. *Journal of Information Security*, *09*(03), 177–190. https://doi.org/10.4236/jis.2018.93013
3. Blockchain for Real Estate Industry. (2019). *The Journal of Social Sciences Research*, *52*, 53–56. https://doi.org/10.32861/sr.52.53.56
4. Garcia-Teruel, R. M. (2020). Legal Challenges and Opportunities of Blockchain Technology in the Real Estate Sector. *Journal of Property, Planning and Environmental Law*, *12*(2), 129–145. https://doi.org/10.1108/jppel-07-2019-0039
5. Dabbagh, M., Sookhak, M., & Safa, N. S. (2019). The Evolution of Blockchain: A Bibliometric Study. *IEEE Access*, *7*, 19212–19221. https://doi.org/10.1109/access.2019.2895646
6. Bhanushali, D., Koul, A., Sharma, S., & Shaikh, B. (2020). BlockChain to Prevent Fraudulent Activities: Buying and Selling Property Using BlockChain. https://doi.org/10.1109/icict48043.2020.9112478
7. Wouda, H., & Opdenakker, R. (2019). Blockchain Technology in Commercial Real Estate Transactions. *Journal of Property Investment & Finance*, *37*(6), 570–579. https://doi.org/10.1108/jpif-06-2019-0085
8. Ølnes, S., Ubacht, J., & Janssen, M. (2017). Blockchain in Government: Benefits and Implications of Distributed Ledger Technology for Information Sharing. *Government Information Quarterly*, *34*(3), 355–364. https://doi.org/10.1016/j.giq.2017.09.007

9. Saari, A., Vimpari, J., & Junnila, S. (2022). Blockchain In Real Estate: Recent Developments and Empirical Applications. *Land Use Policy, 121,* 106334. https://doi.org/10.1016/j.landusepol.2022.106334
10. Morena, M., Truppi, T., Pavesi, A. S., Cia, G., Giannelli, J., & Tavoni, M. (2020). Blockchain and Real Estate: *Dopo di Noi* project. *Property Management*, *38*(2), 273–295. https://doi.org/10.1108/pm-01-2019-0005
11. Veuger, J. (2017). Trust in a Viable Real Estate Economy With Disruption and Blockchain. *Facilities*, *36*(1/2), 103–120. https://doi.org/10.1108/f-11-2017-0106
12. Spielman, A. (2016). Blockchain : Digitally Rebuilding the Real Estate Industry. https://dspace.mit.edu/handle/1721.1/106753
13. Konashevych, O. (2020). General Concept of Real Estate Tokenization on Blockchain. *European Property Law Journal*, *9*(1), 21–66. https://doi.org/10.1515/eplj-2020-0003
14. Gupta, A., Rathod, J., Patel, D., Bothra, J., Shanbhag, S., & Bhalerao, T. (2020). Tokenization of Real Estate Using Blockchain Technology. In *Springer eBooks* (pp. 77–90). https://doi.org/10.1007/978-3-030-61638-0_5
15. Alam, K. M. R., Rahman, J. A., Tasnim, A., & Akther, A. (2020). A Blockchain-based Land Title Management System for Bangladesh. *Journal of King Saud University - Computer and Information Sciences*, *34*(6), 3096–3110. https://doi.org/10.1016/j.jksuci.2020.10.011
16. Thamrin, R. M., Harahap, E. P., Khoirunisa, A., Faturahman, A., & Zelina, K. (2021). Blockchain-based Land Certificate Management in Indonesia. *ADI Journal on Recent Innovation (AJRI)*, *2*(2), 232–252. https://doi.org/10.34306/ajri.v2i2.339

17. Aquib, M., Dhomeja, L. D., Dahri, K., & Malkani, Y. A. (2020). Blockchain-based Land Record Management in Pakistan. In *2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*. https://doi.org/10.1109/icomet48670.2020.9073927

18. Zaman, Q., Idrees, M., Ashraf, A., & Ahmad, A. (2022). A Smart Contract Approach in Pakistan Using Blockchain for Land Management. *International Journal of Innovations in Science and Technology*, *4*(2), 425–435. https://doi.org/10.33411/ijist/2022040213

19. Racetin, I., Kilić, J., Zrinjski, M., & Peko, M. (2022). Blockchain-Based Land Management for Sustainable Development. *Sustainability*, *14*(17), 10649. https://doi.org/10.3390/su141710649

APPENDICES

Appendix A – Procedure

1. Registration and Management

• Visit https://blockestate-gamma.vercel.app to access the website.

• To connect with the Metalmark extension, click the "Connect" button.

• Verify the connection by authenticating yourself with your wallet address.

• Continue the registration procedure.

• Type your CNIC (Computerized National Identity Card) number in the box provided.

• Share your contact and personal information.

• To finish registering, click the "Submit" button.

• You will be led to the Dashboard after registering.

On the "My Account" tab, users may easily manage their contact information and personal information. Their Ethereum wallet address and CNIC are the only account details that they cannot edit in this section. Users can update their name, address, phone number, email address, and other pertinent contact information on this page.

2. Buying and Selling

Users can browse through available properties for sale, each with pertinent information, on the "Marketplace" website. Users are directed to submit a bid request by clicking the "Buy" button on a property card.

The offer made by a prospective buyer is up to the seller to accept or reject. Both the buyer and the seller will digitally sign a contract if the seller accepts the offer.

The buyer must conduct a bank transfer to the seller in order to complete the deal. After that, as proof of payment, the buyer must send a photo or PDF of the bank document. Both the buyer and the seller confirm receipt of the payment, and the system checks the payment proof. A gas fee will be deducted from both the seller's and buyer's wallets. Once the transaction is completed, the land ownership will be transferred, and a legal document with the signatures (wallet addresses) of both parties will be available for viewing and downloading.

Appendix B – Libraries/Packages

- **"@metamask/detect-provider": "^2.0.0":** A library for detecting the provider (e.g. MetaMask) of a user's web3 wallet in a browser.
- **"@openzeppelin/contracts": "^4.8.0":** A library that provides secure and tested smart contracts for building decentralized applications (dApps) on Ethereum.
- **"@prisma/client": "^4.9.0":** An ORM (Object Relational Mapping) library for TypeScript and JavaScript, used to interact with databases from Node.js.
- **"@truffle/contract": "^4.6.8":** A library for interacting with Ethereum smart contracts from JavaScript or TypeScript, providing a simplified way to handle contract creation, deployment, and interaction.
- **"bcryptjs": "^2.4.3":** A library for hashing passwords using the bcrypt algorithm, providing a secure way to store and verify passwords.
- **"crypto-js": "^4.1.1":** A library that provides a collection of cryptographic algorithms and tools for encryption and decryption in JavaScript.
- **"jsonwebtoken": "^8.5.1":** A library for generating JSON Web Tokens (JWTs), used for secure authentication and authorization in web applications.
- **"jspdf": "^2.5.1":** A library for generating PDF documents in JavaScript, providing a range of features for customizing and styling PDFs.
- **"leaflet": "^1.9.3":** A library for building interactive maps in JavaScript, providing a lightweight and customizable solution for displaying map data.

- **"react-icons": "^4.7.1":** A library that provides a wide range of free and open-source icons for React applications.
- **"react-leaflet": "^4.2.0":** A library that provides React components for building maps using the Leaflet library.
- **"react-toastify": "^9.1.2":** A library for displaying toast notifications in React applications, providing a simple and customizable way to show messages to users.
- **"uuidv4": "^6.2.13":** A library for generating UUIDs (Universally Unique Identifiers) in JavaScript, providing a way to generate unique.
- **"web3": "^1.8.1":** A JavaScript library for Ethereum blockchain interaction and dApp development.
- **"tesseract":** Tesseract is used for extracting text from images or scanned documents.

Appendix C - Codes

1. Land token:

```
constructor() ERC721("LandToken", "LTT") {}

function safeMint(address to, uint256 tokenId, string memory uri) public {
_safeMint(to, tokenId); _setTokenURI(tokenId, uri);

address[] memory emptyArray;
bytes32 ownerToken = hashTokenAddress(to,tokenId);
TokenOwner[ownerToken] = TokenOwnerStruct(tokenId,false,emptyArray);
}

function sharedOwnerList(address owner, uint256 tokenId) public view returns(address[] memory) {
bytes32 ownerToken = hashTokenAddress(owner,tokenId);
return TokenOwner[ownerToken].sharedOwners;
}


function setSharedOwners(address owner, address sharedOwner, uint256 tokenId) public {
require(_isApprovedOrOwner(_msgSender(), tokenId), "ERC721: caller is not token owner or approved");

bytes32 ownerToken = hashTokenAddress(owner,tokenId);

bytes32 sharedToken = hashTokenAddress(sharedOwner,tokenId);
TokenOwner[ownerToken].sharedOwners.push(sharedOwner);
TokenSharedOwner[sharedToken] = TokenSharedOwnerStruct(tokenId,true,owner);

}
```

```
function deleteSharedOwners(address owner, address sharedOwner, uint256 tokenId) public {
require(_isApprovedOrOwner(_msgSender(), tokenId), "ERC721: caller is not token owner or approved");

bytes32 ownerToken = hashTokenAddress(owner,tokenId);
address[] memory sharedOwners = TokenOwner[ownerToken].sharedOwners;

bytes32 sharedToken = hashTokenAddress(sharedOwner, tokenId);
TokenSharedOwner[sharedToken] = TokenSharedOwnerStruct(10000000000,false,address(0));

address[] memory outputArray = new address[](sharedOwners.length-1);

for(uint i = 0; i < sharedOwners.length; i++) {
uint j = 0;
if (sharedOwners[i] != sharedOwner) {
outputArray[j] = sharedOwners[i];
j++;
}
}

TokenOwner[ownerToken].sharedOwners = outputArray;
}
```

2. Login

```
async function handler(req, res) {
const { address, sessionStatus } = req.body

const checkSessionQuery = await prisma.$queryRaw(Prisma.sql`SELECT * FROM sessions WHERE address =
${address.toLowerCase()} AND status = 'Active'`)

if(!checkSessionQuery.length>0 && address && sessionStatus === 1) {

const sessionID = uuid()
const salt = bcryptjs.genSaltSync(10)
const key = bcryptjs.hashSync("BlockEstate", salt)

const token = jwt.sign({ sessionID, address }, key)

const insertSessionQuery = await prisma.sessions.create({
data: {
address: address,
session_id: sessionID,
token: token,
status: 'Active',
},
})
```

```
res.json({
sessionID,
address,
token,
status: "Active"
})
}
else if(!checkSessionQuery.length>0 && address) {
res.json(null)
}
else {
res.json({
sessionID: checkSessionQuery[0].session_id,
address: checkSessionQuery[0].address,
token: checkSessionQuery[0].token,
status: "Active"
})
}
```

3. My Properties

```
const MyPropertiesComp = ({ address, web3Api, userDetails, apiKey }) => {

const [userLands, setUserLands] = useState([])

useEffect(() => {
const getAllUserTokens = async () => {
const { landToken, web3 } = web3Api

const eventTokenIds = await landToken.getPastEvents('Transfer', {
filter: {
'to': address,
},
fromBlock: 0,
toBlock: 'latest'
})

const notOwnedTokenIds = await landToken.getPastEvents('Transfer', {
filter: {
'from': address,
},
fromBlock: 0,
toBlock: 'latest'
})

const commonTokenIds = eventTokenIds.filter((event) => {
return !notOwnedTokenIds.some((notOwned) => notOwned.returnValues.tokenId ===
event.returnValues.tokenId);
})
```

```
const lands = []
const promises = []

commonTokenIds.forEach((tokenId) => {
promises.push(
landToken.tokenURI(parseInt(tokenId.returnValues.tokenId))
.then((landTokenURI) => {
lands.push(JSON.parse(landTokenURI))
})
)
})

Promise.all(promises)
.then(() => {
setUserLands(lands)
})
}

web3Api.web3 && getAllUserTokens()

}, [web3Api.web3 && address])
```

4. Web Map

```
import "leaflet/dist/leaflet.css"
import { MapContainer, TileLayer, Polygon, Popup } from "react-leaflet"

const Map = ({ userLands, otherUserLands }) => {
return (
<div>
<MapContainer center={[33.64498558968215, 72.98832287301876]} zoom={13}>
<TileLayer
attribution='&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors'
url="https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png" />

{userLands ? userLands.map(shape => {return shape ?
<Polygon pathOptions={{color:'purple'}} positions={shape.geometry.coordinates}>
<Popup>Hello</Popup>
</Polygon>
: null}) : null}

{otherUserLands ? otherUserLands.map(shape => {return shape ?
<Polygon pathOptions={{color:'red'}} positions={shape.geometry.coordinates}>
<Popup>I am someone else's land</Popup>
</Polygon>
: null}) : null}

</MapContainer>
```

```
</div>
)
}
export default Map
```

5. Land Transfer

```
const TransactionComp = ({ address, web3Api, userDetails, router, apiKey }) => {

const date = new Date().toLocaleDateString()

const [refreshStatus, setRefreshStatus] = useState(false)

const [transactionId, setTransactionId] = useState()
const [transaction, setTransaction] = useState({})

const [seller, setSeller] = useState({})
const [buyer, setBuyer] = useState({})

const [userStatus, setUserStatus] = useState("")

useEffect(() => {
if(!router.isReady) return
setTransactionId(decrypt(router.query.id))
}, [router.isReady])

useEffect(() => {

const transactionDetails = async () => {

const response = await fetch("api/transaction/getTransactionById", {
method: "POST",
body: JSON.stringify({ transactionId, apiKey }),
headers: {
'Content-Type': 'application/json'
}
})

const data = await response.json()

if(!data) return

setTransaction(data.transaction)

}

transactionId && address && transactionDetails()
```

```
}, [transactionId && address, refreshStatus])

useEffect(() => {

const setBuyerSeller = async () => {
if(transaction.seller_address === address.toLowerCase()) {
setUserStatus("Seller")
setSeller(userDetails)

let address = transaction.buyer_address

const userResponse = await fetch("api/user/getUserDetails", {
method: "POST",
body: JSON.stringify({ address, apiKey }),
headers: {
'Content-Type': 'application/json'
}
})

const userData = await userResponse.json()
if(!userData) return
setBuyer(userData)
}
else if(transaction.buyer_address === address.toLowerCase()) {
setUserStatus("Buyer")
setBuyer(userDetails)

let address = transaction.seller_address

const userResponse = await fetch("api/user/getUserDetails", {
method: "POST",
body: JSON.stringify({ address, apiKey }),
headers: {
'Content-Type': 'application/json'
}
})

const userData = await userResponse.json()
if(!userData) return
setSeller(userData)
}
}

transaction && setBuyerSeller()


}, [transaction.id])

const signDocument = async () => {

const response = await fetch("api/transaction/signDocument", {
```

```
method: "POST",
body: JSON.stringify({ transactionId, userStatus, apiKey }),
headers: {
'Content-Type': 'application/json'
}
})

const data = await response.json()

if(!data) return

toast.success(data.message, {
position: toast.POSITION.BOTTOM_RIGHT
})

setRefreshStatus(!refreshStatus)

}

const confirmPayment = async () => {

let landId = transaction.land_id
let sellerAddress = transaction.seller_address
let buyerAddress = transaction.buyer_address
let acceptedPrice = transaction.accepted_price

let bidStatus = "3"

const updateBid = await fetch("api/transaction/updateBidStatus", {
method: "POST",
body: JSON.stringify({ landId, sellerAddress, buyerAddress, acceptedPrice, bidStatus, apiKey }),
headers: {
'Content-Type': 'application/json'
}
})

const updateData = await updateBid.json()
if(!updateData) return

const { landToken } = web3Api

const documentData = {
tokenId: transaction.land_id,
buyer: transaction.buyer_address,
seller: transaction.seller_address,
price: transaction.accepted_price,
date: date
}

const documentDataJSON = JSON.stringify(documentData)
```

```
const landTransferContract = await landToken.transferAgreementDocument(documentData.tokenId,
documentData.buyer, documentData.seller, documentDataJSON, {from:address})

if(landTransferContract) {
toast.success("Transfer complete!", {
position: toast.POSITION.BOTTOM_RIGHT
})
}

}

const payPayment = async () => {

let id = transaction.id

const payPayment = await fetch("api/transaction/payPayment", {
method: "POST",
body: JSON.stringify({ id, apiKey }),
headers: {
'Content-Type': 'application/json'
}
})

const paymentData = await payPayment.json()
if(!paymentData) return

toast.success(paymentData.message, {
position: toast.POSITION.BOTTOM_RIGHT
})

setRefreshStatus(!refreshStatus) }
```