Blockchain Ticketing system

Adarsh Narasimha Murthy, Nidhi Tholar Kuchur, Sachin Pothukuchi, Shreyas Kulkarni
Computer Engineering Department
San José State University (SJSU)
San José, CA, USA
Email: {adarsh.narasimhamurthy, nidhi.tholarkuchur, sachin.pothukuchi, shreyas.kulkarni}@sjsu.edu

***Abstract*—With the advent of the internet, event ticketing has undergone rapid evolution, and most tickets are now sold online. Digital ticketing systems provide an integrated platform to consolidate ticketing, check-in, and analytics, in addition to providing a range of services such as streamlining price changes according to the ticket classes, and multiple payment options. However, this digital transformation has not been without its drawbacks. Tickets bought online can easily be forged and sold to many customers. Some users use bots to purchase event tickets in bulk and resell them at a higher price, a practice called ticket scalping. All this has led to the development of unregulated secondary markets, with the organizers losing out on additional profits and customers unable to authenticate their purchases.**

In this project, we propose an NFT (Non-Fungible token) based ticketing system that allows us to regulate the purchase of tickets and provides a mechanism to authenticate tickets to event organizers and attendees. NFTs represent unique digital assets stored in the blockchain with only one owner at any point in time. Since event tickets are valid for a particular seat in the event, tickets can be considered NFTs. Every transaction of the NFT resides immutably in the blockchain. The event organizers can track the resale of tickets and also set limits on the pricing and number of resales. Since all the tickets reside in the Blockchain as NFTs, it acts as a decentralized trusted source to prove the ticket's authenticity. Using NFTs as tickets intimately connects the organizers with the actual attendees, enabling them to offer them exclusive benefits for future or current events.

***Index Terms*—blockchain, NFT, event ticketing, Ethereum, smart contracts, Polygon**

I. Introduction

Events are organized frequently in various fields, such as education, entertainment, and social awareness. While some events are open to all, most require customers to have valid tickets. Event ticketing has rapidly evolved over the years, with most tickets now being sold online. Digitization has made ticket management easy both for organizers and ticket buyers. Organizers can vary ticket prices based on demand and connect with users by sending event-related communications. Organizers can also easily track the number of attendees to manage the events better. Ticket buyers can buy tickets from the comfort of their homes. Digital wallets and QR codes have made check-ins and other ticket-related transactions seamless. However, digitized ticketing systems come with drawbacks. Tickets bought online can be counterfeit and sold to more than one customer. Tickets are also purchased in bulk and resold at a higher price. This kind of fraud has made it difficult to authenticate tickets. It also creates a barrier between organizers and users. Hence, there is a need for a more secure ticketing system.

Using blockchain technology can enhance the security aspect of ticketing systems. Blockchain is a chain of blocks that stores a record of transactions. Each block comprises a header and a body uniquely identified by a hash. The header links the blocks by storing the preceding block's hash, forming an unalterable chain. Since blockchain is decentralized, it does not require a central authority or a trusted third party to verify transactions. Smart contracts are self-executing codes that automate and customize transactions on the blockchain, enabling the execution of agreements between buyers and sellers.

The Blockchain's traceability feature makes it preferable to traditional systems, particularly in supply chain management [1]. By utilizing a blockchain-based ticketing system, organizers and entities can authenticate ticket ownership and track ticket resale activities, thanks to the immutable records stored for every step of the ticket-buying process.

Non-fungible tokens (NFTs) can be integrated with blockchain technology to represent digital tickets. NFTs are unique digital assets with only one owner at a time, and their ownership can be traced on the blockchain. Furthermore, NFTs can be customized based on the event, adding value to the tickets and making them collectibles.

We propose an application that leverages Ethereum blockchain technology with Polygon as a layer 2 scaling solution to enhance event ticketing. Our solution utilizes blockchain and NFTs to provide secure, transparent, and traceable ticket transactions.

II. Related Work

A. Blockchain

S. Nakamoto introduced blockchains in his seminal paper on Bitcoins [2]. As the name suggests, a blockchain is a list of blocks, with each block containing a block header and a Merkel Tree of transactions. A new block can be added to the chain by a consensus mechanism, depending on the type of blockchain. The consensus mechanism used in bitcoins is Proof of Work (PoW). In the PoW scheme to add a new block, we need to generate a nonce that can make the hash value of a block with a certain number of zero bits. A node with higher computing power will have a higher chance of generating the required nonce.

S. King and S. Nadal introduced another consensus mechanism called Proof of Stake(PoS) [3]. With PoS, nodes stake a certain amount of cryptocurrencies for the right to add a new block to the network. The blockchain selects the validator node based on the amount and age of the coins staked. The advantage of PoS is that it needs less computing power and is more scalable.

D. Larimer developed an advanced version of the PoS mechanism called delegated Proof-of-stake(DPoS) [4]. In the case of DPoS, all nodes that stake to validate a transaction can participate in the process. DPoS uses a voting mechanism to decide who will be the validator nodes. This results in a more efficient blockchain.

According to K. Wüst and A. Gervias[5], there are two types of blockchain based on the rules for nodes to join the network. The most common one is the public blockchain, where anyone can become a new node and add/read transactions to the blockchain. Bitcoin and Ethereum are popular examples of this. The other type is the permissioned blockchain, where only some nodes can become part of the blockchain. Hyperledger, Quorum, and Corda are some examples of this type of blockchain.

Ethereum, developed by V. Buterin [6], was the first blockchain that supported virtual machines with Smart contracts written in Turing complete languages. To nurture interoperability, the ERC-20 (Ethereum Requests for Comments) standard by F.Vogelsteller[7] standardized the interface for fungible tokens, which has led to its use in various fields such as banking, real estate, healthcare, sports, and supply chain management. Here, fungibility refers to the interchangeability of a commodity with another. In the 2018 ERC-721 standard, W. Entriken et al.[8] introduced a new type of token called non-fungible tokens(NFTs). While fungibility is an essential feature of any currency, in the case of non-fungibility, every token is distinct with a unique ID.

The introduction of NFTs built a solid use case for its adoption in Event ticketing to resolve many issues with the current system, such as ticket counterfeiting, price gouging, ticket scalping, and control of secondary markets. These issues can be resolved using event tickets as NFTs on the blockchain.

B. Event Ticketing

A ticket is a token that provides access to an event. It comes in many forms, such as physical paper, QR codes, or barcodes. Tickets can be bought from the event organizer or authorized sellers. Secondary markets also exist where tickets sell at highly elevated prices. Customers have to trust unverified sellers in secondary markets and thus face the risk of fraud. Using QR codes or barcodes does not solve this issue, as these can be duplicated. According to Waterson [9], an estimated 12% of ticket buyers get scammed.

There are many solutions to control resale and fraud. The most common way is to print the ticket holder's name on the ticket and identify his identity at the event entrance. But performing these checks is costly and time-consuming [9]. Other approaches propose dynamic QR codes[10] that require account login at the venue. Yet, we can bypass these systems using bots.

C. Blockchain-Based Event Ticketing Systems

We propose to use blockchain to improve ticket validation and resale. B. Tackmann[11] developed a prototype with tickets as assets on the Hyperledger Fabric blockchain, secured with digital signatures. Ticket creation, sale, validation, resale, and withdrawal are recorded on the Blockchain, preventing many of the issues mentioned before. However, the proposed solution does not place controls on ticket resales and lacks an explanation about payments. J.D Preece and J.M Easton[12] also proposed a ticketing system utilizing IBM's Hyperledger Fabric framework but in the context of railway ticketing. They decided to restrict access to the transactions by using separate channels for each vendor organization. While this is apt for Railway ticketing, it is unsuitable for Event ticketing, where the ticket buyer can become the reseller. K. Lin et al.[13] developed a blockchain mobile ticketing system on EOSIO, a Proof-of-Stake(PoS) blockchain. It uses the digital signatures of the event organizer and ticket buyer to validate the tickets but does not have a mechanism for resale.

S. -C. Cha et al.[14] proposed a Non-Interactive Zero-knowledge(NIZK) scheme to protect user privacy and allow users to prove their identity without revealing identifiable information. The NIZK scheme uses an elliptic curve cryptography public and private key pair. S. Feulner et al.[15] explored SSI (self-sovereign identity) to resolve user identity. With SSI, users store identity-related documents in digital wallets on their smartphones.

T. Lee, Y. Kim, and J. Yo[16] presented a mechanism for a trusted reselling service. They proposed that the seller and buyer make collateral deposits, which they will lose in case of dishonest behavior by either party. It also requires the user to have a cryptocurrency wallet called MetaMask to establish his identity. P. Corsi, G. Lagorio, and M. Ribaudo[17] define an open specification called TickEth. It also utilizes digital documents such as passports to establish user identity and prevent bots from bulk-buying tickets. The event organizers control the secondary markets by placing limits on the resale price range and the number of resales.

III. Proposed Solution

Digital ticketing systems, despite their capabilities, remain vulnerable to fraud. There is a growing need for improved security and transparency in ticketing systems. Our proposed solution employs blockchain technology, as it offers the necessary security and traceability for ticketing systems. The blockchain ledger maintains a record of each transaction, enabling us to trace ownership and authenticate their validity. Furthermore, the ledger is immutable, making the system highly trustworthy. By leveraging non-fungible tokens (NFTs), we can fully streamline the blockchain-based ticketing system, making the tickets more valuable.

IV. Project Design

Our project design comprises two primary components. The first component is a web application that users can utilize to engage with the system. The second component is the

Blockchain module, which implements the application's core functions.

The web application will provide organizers with the capability to create event tickets, while customers can browse available events and purchase tickets. The blockchain module will facilitate the creation and transfer of non-fungible tokens (NFTs) to customers upon the successful completion of a purchase.

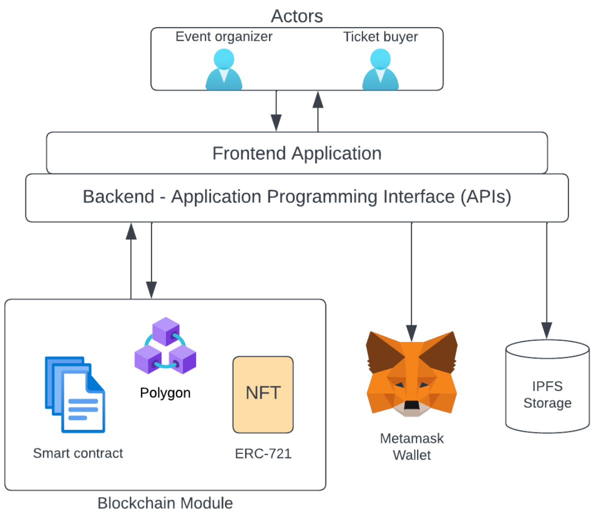


Fig. 1. System Architecture of proposed event ticketing system

Fig. 1 describes the system architecture. As presented in the diagram, our solution has the following components:

Actors: The proposed application involves two actors: the event organizer and the ticket buyer. There is no distinction in user privileges, meaning that a user can function as both an event organizer and a ticket buyer.

Application front-end: To ensure high responsiveness, we chose Next JS to develop the front end of the application.

Application back-end: The backend was developed using Node.js and Web3.js. The backend is responsible for connecting the front end with the blockchain module. The backend receives HTTP requests from the client, triggering the corresponding functions on the blockchain module and providing the necessary response to the client.

Blockchain: The NFT ticketing application was initially developed using the Ethereum blockchain, leveraging the ERC721 standard for NFTs to ensure the security and immutability of ticket ownership. Later, the application was scaled by leveraging the Polygon blockchain for improved scalability and reduced gas fees.

Smart Contracts: Self-executing codes called smart contracts are used to customize and automate the transactions on the blockchain [11]. Smart Contracts are deployed on the blockchain and utilize decentralized computing power to enforce the terms of the contract. They can be defined using programming languages such as Solidity, Vyper, and Rust. We used Solidity to develop our Smart Contract due to its rich library, tooling, and documentation. Solidity is used in EVM-compatible blockchains such as Ethereum, Polygon, Immutable X, Avalanche, Binance, and Tron. Using Solidity proved to be a wise choice, as we later switched to the Polygon blockchain.

Metamask Wallet: Metamask is a secure wallet that stores crypto coins. It also provides a plugin using which we can connect to marketplaces. Users should create an account on Metamask to perform transactions on our application. Users can access their meta mask through a browser extension.

Hardhat: We use hardhat as a development environment as it has inbuilt functionality to start a local blockchain, compile our contract code and, deploy our contract to different blockchains i.e., a local chain, test networks, and mainnet.

IPFS: The Interplanetary File System (IPFS) is a distributed file storage that allows computers all over the globe to store and serve files as part of a giant peer-to-peer protocol. We use IPFS to store NFT images and metadata related to tickets such as event name, description, and date.

V. Project Implementation

This section of the report discusses the implementation details of our proposed solution. The section is further divided into the following parts: Blockchain, Smart Contracts, NFTs, and Web applications.

A. Blockchain

The blockchain module is at the core of our application, and its reliability and scalability are critical for ensuring a smooth user experience. During our initial research and experiments, we identified the Ethereum blockchain as a promising platform for our ticketing system, given its security and reliability features. However, as we continued to test our application, we encountered challenges with Ethereum's limited scalability, including slow transaction speeds and high fees during periods of network congestion. To address these issues, we explored potential solutions, including Layer 2 scaling solutions. Layer 2 solutions are designed to increase the transaction processing capacity of a blockchain, and we ultimately chose to use Polygon, a Layer 2 solution compatible with Ethereum. Polygon employs a sidechain architecture that offloads some of the computational work from the Ethereum mainnet, resulting in significantly faster and cheaper transactions. Below we discuss in detail the blockchain implementation of our application.

a) Ethereum blockchain: Ethereum's decentralized architecture and advanced capabilities have established it as a highly sought-after blockchain platform with diverse applications. Our initial experiments found Ethereum to fit our Ticketing system well. We used the Hardhat development environment for Ethereum and wrote smart contracts in Solidity. The smart contracts created NFT tickets for events and transferred ownership of NFT on ticket purchases. To assess the scalability of our blockchain, we conducted load testing by writing REST wrappers around our smart contracts to simulate multiple concurrent transactions, executing 1000 transactions concurrently. However, we found the results unsatisfactory, as the Ethereum blockchain could only support 35 transactions per second, however, our application could only achieve 9 transactions per second. To improve the scalability of the blockchain, we decided to explore Layer 2 solutions for Ethereum.

b) Ethereum Layer 2 solution - Polygon: Polygon, formerly known as Matic Network, is a Layer 2 scaling solution built on top of the Ethereum blockchain. It provides faster and cheaper transactions for users and developers by creating side chains that are integrated with the Ethereum chain later. Polygon's modular and flexible architecture enables customization of blockchain-based applications, and its Layer 2 scaling solution enables high transaction throughput and lower transaction fees, improving the overall user experience.

To leverage Polygon's scalability, we rewrote our smart contracts in Solidity to be compatible with Polygon. Our load testing, which used REST wrappers around our smart contracts, showed that Polygon could support up to 875 transactions per second, a significant improvement over Ethereum's maximum throughput. Furthermore, the gas fees on Polygon were significantly lower than on Ethereum, making it more cost-effective for users to purchase NFT tickets. With the migration to Polygon, we achieved the scalability required for our ticketing system while still leveraging the security and decentralization of the Ethereum blockchain.

B. Smart Contracts

The Ethereum Virtual Machine (EVM) is supported by Ethereum and Polygon, which makes it easy to migrate Smart contracts written in Solidity for Ethereum to Polygon. Since Polygon is designed to address Ethereum's scalability issues, smart contracts written in Solidity for Ethereum can be modified to function on Polygon with minimal changes.

For our NFT ticketing application, we used Hardhat as our development environment to write smart contracts in Solidity for Ethereum. When we decided to explore Polygon as a Layer 2 scaling solution, we made the necessary adjustments to our smart contracts to ensure compatibility with Polygon. This involved modifying the contracts to work with Polygon's specific blockchain parameters and updating the contract addresses to reflect the migration to Polygon. With these changes in place, we could deploy the same smart contracts to the Polygon network and take advantage of Polygon's scalability benefits while still using the same smart contract code that we developed for Ethereum.

Algorithm 1 NFT creation

```
OrganizerEAddress: Seller's Ethereum address
ticketMetadata: ticket details

Input: OrganizerEAddress, ticketMetadata, ticketCount
Generate 'ticketCount' number of tokens
Call mintNFT(token, OrganizerEAddress)
Send NFT creation success message
Return: token
```

Algorithm 2 NFT ownership transfer

```
BuyerEAdress: User's (ticket buyer) Ethereum address

Input: BuyerEAdress, token
Call transferNFT(token, BuyerEAdress)
Send NFT transfer success message)
Exit
```

C. RPC Provider

After migrating our smart contracts to Polygon, we needed a way to interact with the blockchain. We chose to use Infura.io and Alchemy as our RPC providers because they allowed us to connect to the Polygon network and communicate with our smart contracts without setting up our node infrastructure. This saved us time and resources while providing reliable and scalable node infrastructure. By relying on Infura.io and Alchemy, we were able to focus on developing and testing our smart contracts while ensuring a fast and reliable experience for our users. Overall, Infura.io and Alchemy helped us simplify the complexities of connecting to the blockchain and allowed us to seamlessly interact with the Polygon network.

D. Non-fungible tokens

Non-fungible tokens (NFTs) represent tickets in our application, and the NFTs follow ERC-721, an industry standard. Each NFT is unique; for this purpose, we use the Ticket metadata, specifically the tokenId of a ticket. Ticket metadata includes tokenID, eventID, and a hash. The hash includes the owner's details and ticket URI. If the owner's address is empty/null, the ticket hasn't been sold yet.

Our smart contract deployed on the blockchain has the following functionality:

- To create new tickets for a certain event. An event host can come on the website to create a new set of tickets for a particular event, once these tickets have been created, with the relevant image, price, description, and name, they are hosted on from our smart contract based on the listing price that we specify.
- Each ticket can then be bought by a user, they pay the sale price of the ticket and the ownership of the ticket is transferred to them.
- Each user also has the ability to resell the ticket back to be listed again, the user gets their payment back but has to pay the listing price again.
  The smart contract and all of the data regarding the tickets is hosted on the blockchain itself. This lets us have a ticketing system that is based purely on the blockchain itself.

E. Web Application

The web application was built so the user can easily interact with our Blockchain-based ticketing system. The backend is built on Node.js and Web3.js to communicate with the blockchain, and the front end, built on React.js serves as the visual interface.

a) Application Front End: The web interface comprises of following Features: Event Hosting: Users can complete a simple form to create a new event. Event details such as name, description, location and date, and time must be entered along with the number of tickets available for purchase. The User's Metamask credentials will be used to validate the eevent creation process.

Event Listing: This serves as the Home page of our application and will have a collection of Event cards displaying the important Event details. Here, users can browse through events and proceed to click on an event they would like to purchase a ticket.

Event Card: There will be two types of Event Cards, one that displays basic information on the home page and another that will have all necessary Event details filled in by the event organizer for the Event details page.

Event Details and Ticket Purchase: The users are directed to this Page when they click on an event card on the Home Page. This page will display all event details and provide an option to purchase tickets.

User profile: This page will allow the user to view his Metamask wallet through the browser section. This will allow the user to view the Tickets he/she owns

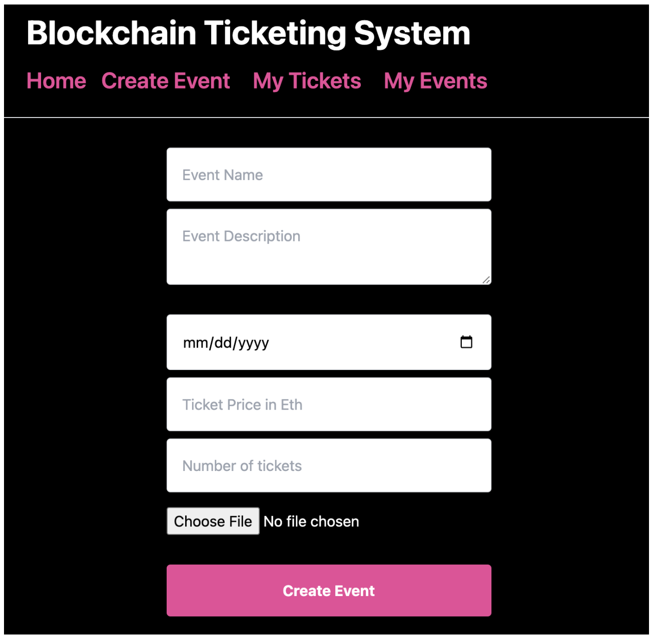


Fig. 2. Create Event page

b) Application Backend: The Backend comprises the blockchain module, which was discussed in detail in the previous sections.

Web3.js: To interact with the smart contracts on the blockchain, the Web3.js library was used. This library provides a way to communicate with the Ethereum node and execute smart contracts functions.

Authentication and Authorization: Metamask is utilized to authenticate and authorize all transactions on the application.

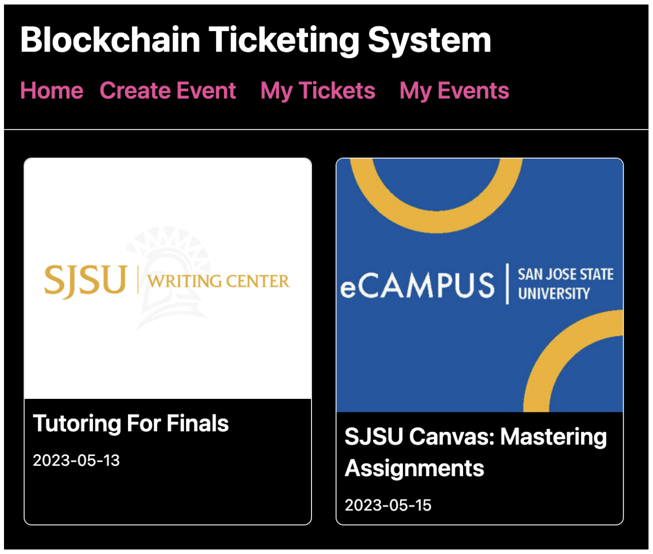
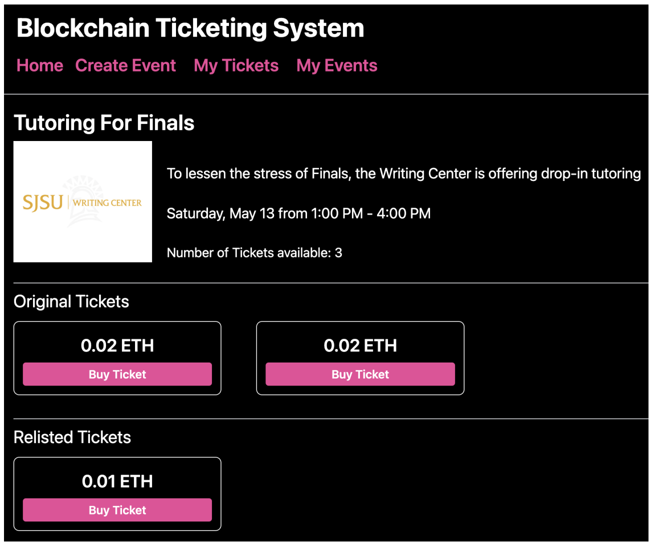

Blockchain Ticketing System
Home
Create Event
My Tickets
My Events
SJSU | WRITING CENTER
eCAMPUS | SAN JOSE STATE UNIVERSITY
Tutoring For Finals
2023-05-13
SJSU Canvas: Mastering Assignments
2023-05-15

Fig. 3. Home Page



To lessen the stress of Finals, the Writing Center is offering drop-in tutoring

Saturday, May 13 from 1:00 PM - 4:00 PM

Number of Tickets available: 3

Fig. 4. Event Details and Ticket Purchase page

It is required that the User logins to his meta mask wallet using the browser extension.

API Development: We developed REST APIs to provide a way for the front-end of the web application to communicate with the backend. To secure the APIs each request was inspected for valid credentials. APIs were written for Event Creation, Ticket purchase, and fetching User profile details.

Data and Storage: In addition to using Ethereum and Polygon for the blockchain infrastructure, we also leveraged IPFS (InterPlanetary File System) for storing and distributing the media files associated with our NFT tickets. IPFS is a decentralized and distributed file storage system that provides secure and efficient file sharing across a peer-to-peer network. By using IPFS, we were able to avoid relying on a centralized server for file storage, which can be costly and prone to single points of failure. Instead, we stored our media files, such as images and videos, on the IPFS network, which

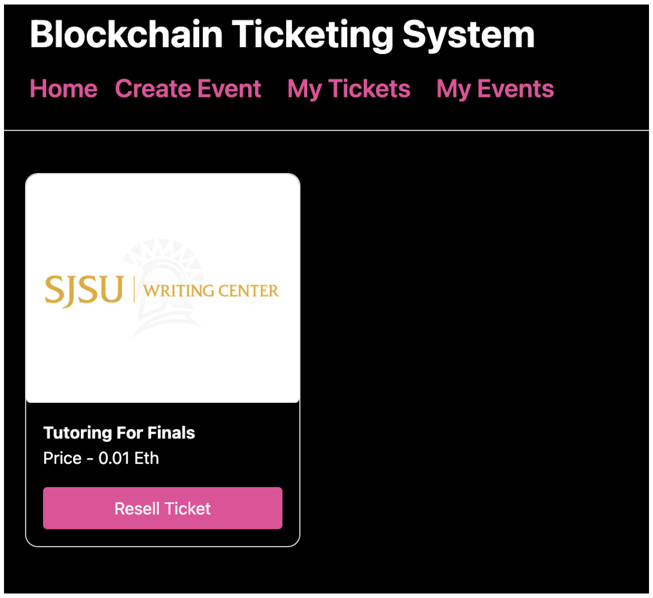


Fig. 5. My Tickets page

allowed for easy distribution of the files to our users. We also used IPFS to ensure that the media files associated with each NFT ticket were immutable and tamper-proof, as any changes made to the files would be detected by the network. By using IPFS, we were able to provide a secure and decentralized file storage solution for our NFT ticketing application.

c) Testing: Two types of tests were performed:

Unit Testing: unit testing was performed using the Chai testing library. Chai is a popular library for testing Web3 applications and is compatible with most of the modern javascript frameworks.

Load testing: Load Testing was performed in Apache Jmeter, which is arguably the most popular tool for performing load testing. Our smart contract was tested for the below parameters on a total of 12000 transactions for 300 users, ramp up-period of 1 second, and a loop count of 40. The transaction size for buying a ticket on our smart contract is 362 bytes.

i) TPS (Transactions per Second): In the realm of blockchain technology, a commonly used metric to gauge the efficiency of a smart contract is transactions per second (TPS), which refers to the number of transactions that can be processed by the contract within a second. It is important to note that TPS solely measures the speed of transaction processing and does not account for the time taken to finalize a transaction. After a transaction is submitted to the blockchain, validators verify its legitimacy, but it may not be added to the blockchain immediately; rather, it may take several blocks to be added. Therefore, TPS does not necessarily reflect the real-world user experience.

ii) TTF(Time to Finality): TTF is analogous to latency in network terminology. TTF is the amount of time needed for a transaction to be written immutably to a blockchain after a legitimate transaction has been submitted. The TTF of a blockchain depends on several factors such as block time, block size, and the consensus mechanism used.

In the context of event ticketing, TTF is important because it ensures that tickets cannot be duplicated or transferred to unauthorized parties. If a ticketing transaction is not finalized and confirmed on the blockchain in a timely manner, it could be subject to fraudulent activities such as double-spending or counterfeit ticketing.

On the other hand, while TPS is still important for event ticketing on the blockchain, it may be less critical than TTF. This is because ticketing transactions tend to be relatively low in frequency compared to other blockchain applications such as cryptocurrency trading or gaming. That being said, it's still important for the blockchain network to be able to process transactions efficiently and quickly in order to ensure that tickets can be purchased and transferred in a timely manner.

iii) Average Gas used: Gas usage is used to measure the computational effort required to execute transactions. This cost is paid in the form of transaction fees. The higher the gas usage, the higher the transaction fees required to execute a transaction or smart contract. If gas usage is too high, it can lead to network congestion and slower transaction processing times.

VI. Results

The smart contract was initially deployed on Ethereum and load testing was performed using the local hardhat testing framework to calculate the TPS for 1000 transactions. As shown in Fig. 6, The average TPS was found to be 9, and this was our motivation to explore layer-2 solutions such as Polygon.

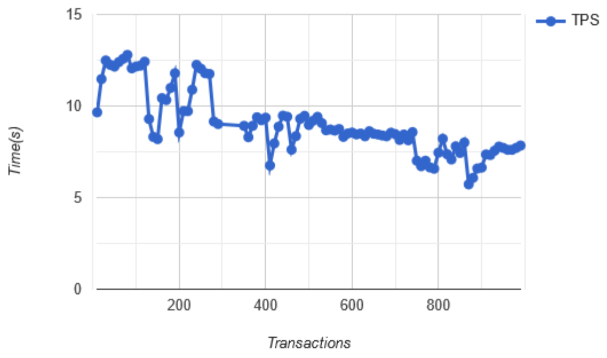


Fig. 6. Load Testing of the Smart Contract on Ethereum for 1000 transactions

Polygon was deployed on testnet, hence we could not use the local hardhat testing framework to perform the tests. Therefore, we created REST wrappers in Node.js to interact with the smart contract. These REST wrappers were called in Apache Jmeter with 1000 virtual users to simulate the load test.

The REST wrappers include functionality to compile gasPrice, gasLimit, TransactionCount, and getwalletAddress, which would be performed by Metamask in a real-time environment. The readings below exclude any overheads caused by these operations. Additionally, since we are load testing through the REST application, and not directly with the smart contract on the blockchain, this may have introduced an additional overhead.

Metric	Ethereum	Polygon
Block Size (in bytes)	121377	55000
Transactions/ block	355	152
Block Time (secs)	13	2
TPS	9	303
TTF(seconds)	15	5

TABLE I
COMPARISION BETWEEN ETHEREUM AND POLYGON

From Table 1, we can see that Polygon is significantly faster than running on the main Ethereum layer. While we were not able to reach the theoretical TPS for either of the blockchains we saw a jump in throughput from 9 transactions per second to 303. As such, using Layer-2 solutions built on top of Ethereum has helped us improve the throughput almost by a factor of 100. We also see a significant drop in TTF from 15 in Ethereum to 5 in Polygon, this helps in the user experience and makes it feel closer to a monolithic architecture in terms of wait times from buying the ticket to receiving the tickets.

VII. CONCLUSION

In conclusion, our NFT ticketing application leveraged the power of blockchain technology to provide a secure and decentralized ticketing solution. We initially experimented with Ethereum and found it to be a suitable fit for our ticketing system. However, as we conducted load testing, we discovered Ethereum's scalability limitations would not meet our application's requirements. To address this, we migrated our smart contracts to the Polygon network, a Layer 2 scaling solution for Ethereum. This allowed us to take advantage of the scalability benefits of Polygon while still offering the security and reliability of Ethereum. We also used Infura.io as our RPC provider to connect to the Polygon network and IPFS to store and distribute the media files associated with our NFT tickets. By leveraging these technologies, we were able to provide a fast, secure, and decentralized ticketing solution for our users. Our NFT ticketing application demonstrates the potential for blockchain technology to transform the ticketing industry and opens up new possibilities for secure and transparent ticketing systems in the future.

ACKNOWLEDGMENT

We would like to thank Dr. Gheorghi Guzun for his support and guidance.

REFERENCES

[1] N. Alnuaimi, A. Almemari, M. Madine, K. Salah, H. Al Breiki, and R. Jayaraman, "Nft certificates and proof of delivery for fine jewelry and gemstones," *IEEE Access*, vol. 10, pp. 101 263–101 275, 2022.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[3] S. King and S. Nadal, "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake," *self-published paper, August*, vol. 19, no. 1, 2012.

[4] D. Larimer, "Delegated proof-of-stake (dpos)," *Bitshare whitepaper*, vol. 81, p. 85, 2014.

[5] K. Wüst and A. Gervais, "Do you need a blockchain?" in *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*. IEEE, 2018, pp. 45–54.

[6] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, vol. 3, no. 37, pp. 2–1, 2014.

[7] F. Vogelsteller and V. Buterin, "Eip 20: Erc-20 token standard," *Ethereum Improvement Proposals*, vol. 20, 2015.

[8] W. Entriken, D. Shirley, J. Evans, and N. Sachs, "Eip-721: Erc-721 non-fungible token standard," *Ethereum Improvement Proposals*, no. 721, 2018.

[9] M. Waterson, "Independent review of consumer protection measures concerning online secondary ticketing facilities," 2016.

[10] L. F. Freitas, A. R. Nogueira, and M. E. V. Melgar, "Data validation system using qr code and meaningless reversible degradation," in *2019 International Conference on Applied Electronics (AE)*. IEEE, 2019, pp. 1–4.

[11] B. Tackmann, "Secure event tickets on a blockchain," in *Data privacy management, Cryptocurrencies and Blockchain technology*. Springer, 2017, pp. 437–444.

[12] J. D. Preece and J. M. Easton, "Blockchain technology as a mechanism for digital railway ticketing," in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2019, pp. 3599–3606.

[13] K.-P. Lin, Y.-W. Chang, Z.-H. Wei, C.-Y. Shen, and M.-Y. Chang, "A smart contract-based mobile ticketing system with multi-signature and blockchain," in *2019 IEEE 8th Global Conference on Consumer Electronics (GCCE)*. IEEE, 2019, pp. 231–232.

[14] S.-C. Cha, W.-C. Peng, T.-Y. Hsu, C.-L. Chang, and S.-W. Li, "A blockchain-based privacy preserving ticketing service," in *2018 IEEE 7th Global Conference on Consumer Electronics (GCCE)*. IEEE, 2018, pp. 585–587.

[15] S. Feulner, J. Sedlmeir, V. Schlatt, and N. Urbach, "Exploring the use of self-sovereign identity for event ticketing systems," *Electronic Markets*, pp. 1–19, 2022.

[16] T. Le, Y. Kim, and J.-Y. Jo, "Implementation of a blockchain-based event reselling system," in *2019 6th international conference on computational science/intelligence and applied informatics (CSII)*. IEEE, 2019, pp. 50–55.

[17] P. Corsi, G. Lagorio, and M. Ribaudo, "Ticketh, a ticketing system built on ethereum," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, 2019, pp. 409–416.