
VILNIUS UNIVERSITY
FACULTY OF MATHEMATICS AND INFORMATICS
INSTITUTE OF COMPUTER SCIENCE
INFORMATION TECHNOLOGIES STUDY PROGRAM

Course work, Information Technology 3rd year

# Decentralized payment and accommodation exchange system using Web3 and Blockchain technologies.

Done by:
Deividas Bendaravičius

Supervisor:
Linas Būtėnas

Vilnius
2023

# Contents

# 1 Abstract

Blockchains, Web3, and decentralization have emerged as prominent and widely discussed topics lately. However, despite their popularity, many users exhibit anxiety towards these innovative technologies, primarily due to their rooted reliance on centralized applications. Concerns regarding data security, system complexity, and potential fraud contribute to the hesitation surrounding the adoption of decentralized applications (Dapps) as substitutes. This project aims to address these concerns by showcasing the user-friendly, transparent, and secure nature of decentralized applications. To achieve this, I developed an accommodation system similar to well-known platforms like Airbnb or Booking, employing Web3 technologies, from user data oversight to payment handling, as well as different ways of imagining the web with real-time payment streaming. The research conducted during the project demonstrates that implementing a user-friendly and intuitive system within a real-life context is both possible and beneficial. Financially, it offers benefits to the user, while surpassing the security standards of traditional centralized applications. In conclusion, this work symbolizes an important development in the progression of decentralized technologies and their transformative potential in reshaping our interaction with digital services. The future appears promising for Web3 technologies and their ability to establish an open, transparent, and equitable digital realm for all.

***Keywords:*** Decentralization, Blockchain, Decentralized applications (DApps), Accommodation, Bookings

# 2 Santrauka

Blokų grandinės, Web3 ir decentralizacija pastaruoju metu tapo populiariomis ir plačiai aptariamomis temomis. Nepaisant jų populiarumo, daugelis vartotojų nerimauja dėl šių inovatyvių technologijų, visų pirma dėl to, kad yra priklausomi nuo centralizuotų programų. Duomenų saugos, sistemos sudėtingumo ir galimų apgavysčių rūpesčiai prisideda prie decentralizuotų programų (DApps), kaip pakaitų, įsikūrimo. Šis projektas skirtas šioms problemoms spręsti demonstruojant vartotojui patogų, skaidrų ir saugų decentralizuotų programų pobūdį. Tai pasiekti, buvo sukurta decentralizuota mokėjimų ir apsikeitimo tipo sistema, kuri vykdo apgyvendinimo funkciją, panašią į gerai žinomas platformas kaip „Airbnb“ arba „Booking“, tačiau naudojant „Web3“ technologijas. Nuo naudotojų duomenų priežiūros iki mokėjimo apdorojimo. Taip pat parodant skirtingus būdus įsivaizduoti naujas internetines sistemas, buvo pasinaudota saugiu mokėjimų srautiniu perdavimu realiuoju laiku. Projekto metu atlikti palyginimai ir sistemos realizacija rodo, kad sukurti vartotojui patogią ir intuityvią sistemą realiame kontekste yra įmanoma ir naudinga. Ne tik finansiškai ji suteikia vartotojui pranašumų, bet tuo pat yra pranašesnė duomenų saugumo atžvilgiu ir standartais nei tradicinės centralizuotos sistemos. Galiausiai šis darbas prisideda prie ir simbolizuoja svarbų decentralizuotų technologijų progresą ir jų transformavimo potencialą, pertvarkant mūsų kaip vartotojų sąveiką su skaitmeninėmis paslaugomis.

# 3 Introduction

Nexstay is an early decentralized accommodation and booking application that acts as a decentralized payment and goods exchange system. Built on the principles of decentralization and blockchain technology, Nexstay offers a secure, transparent, and user-centric platform for booking accommodations.

One of the core features of Nexstay is its ability to handle payments using blockchain technology. Traditionally, when booking accommodations through centralized platforms, users are required to entrust their payment information to third-party intermediaries. This introduces potential security risks and additional transaction fees. In contrast, Nexstay eliminates the need for intermediaries by leveraging the inherent security and transparency of blockchain networks. Payments made on Nexstay are securely processed on the blockchain via smart contracts, ensuring the integrity of transactions and providing users with peace of mind regarding their financial information.

Moreover, Nexstay introduces innovative ways of integrating new technologies in exchanging digital assets for real-world goods not only through standard instant payments but also through its real-time payment streaming functionality. This feature allows for a seamless and continuous flow of funds, enabling users to pay for their accommodations in a more flexible and convenient manner. By leveraging the capabilities of blockchain technology, Nexstay ensures the immediate and secure transfer of digital assets, creating a frictionless experience for both hosts and guests.

In addition to its payment capabilities, Nexstay prioritizes user privacy and data ownership. Unlike traditional centralized platforms that collect and control user data, Nexstay empowers individuals to have complete control over their personal information. User data is stored securely on the blockchain, and users have the ability to grant access to their data selectively, ensuring their privacy is protected.

By combining the benefits of decentralization, blockchain technology, and user-centric design, Nexstay represents an influential solution to traditional centralized applications. It exemplifies the potential of decentralized applications to disrupt traditional models, offering users greater control, security, and transparency in their user and payment data.

As the adoption of decentralized technologies continues to grow, Nexstay serves as a testament to the fundamental power of Dapps. It represents a model shift towards a more decentralized and fair future, where individuals have increased autonomy and ownership over their digital interactions and transactions. The innovative features and principles embodied by Nexstay contribute to paving the way for a new era of decentralized applications, revolutionizing various industries, and empowering users worldwide.

# 4 Web3 Analysis

In this section, I will present the reasoning for Web3 usage, some comparison between the Web2 and the Web3 model in the payments, as well as data security sense. In addition, will provide information on Web3 usage in the project's scope (Authorization process, Payments process).

## 4.1 Reasoning of Web3 usage

Web3, also called the decentralized web, is the next evolution of the web as we know it, which aims to dispose of centralized manipulation and convey back the ownership and control of information to people. This is achieved through using blockchain technology, which lets in for secure, transparent, and decentralized transactions and interactions.

Web3 differs from Web2, the modern-day generation of the web, which is essentially centralized and relies on huge businesses (eg. Microsoft, Google) and intermediaries to manage and, in some cases, sell user data. Web2 has been criticized for its loss of privacy, censorship, and exploitation of user data by using those intermediaries.

More and more websites are moving to Web3 to take gain of its decentralized nature, which offers elevated privacy, security, and control over user data. This shift allows customers to have greater ownership and control over their information and to interact with websites and packages without the want for intermediaries or centralized authorities.

**The main reason why this project uses Web3** - to take a shift and contribute towards a greater democratized and decentralized net, where users will have more control over their records and can interact with websites and programs in an extra secure and transparent manner. Additionally, to show that integrating day-to-day websites with the newest Web3 technologies is not resource intensive, as well as, does not cause the user any harm, as the current tools allow the interactions to be user-friendly.

## 4.2 Transaction benefits and Competitor analysis

Using Web2 payments for accommodation typically requires the use of intermediaries such as banks or payment processors. These intermediaries charge a fee for their services and may also require users to share sensitive financial information, such as credit card numbers, which may put users at risk of fraud or identity theft.

On the other hand, Web3 offers several advantages for payments:

- **Decentralized payments:** With Web3, payments can be made directly between individuals without intermediaries. This reduces transaction fees and makes payments faster and more efficient.
- **Smart Contracts:** Web3 enables the use of smart contracts, which are self-executing contracts where the terms of the contract are written directly into code. It can automate the payment process and ensure that payments are only made when certain conditions are met, such as a residential lease.
- **Cryptocurrency payments:** Web3 also supports the use of cryptocurrencies, which offer several advantages over traditional payment methods. Cryptocurrencies can offer better privacy and security because they are not tied to personally identifiable information like credit

cards. In addition, cryptocurrency payments are often faster and cheaper than traditional payments because they do not require an intermediary to process the transaction.

### 4.2.1 Transaction fee comparison

As mentioned above, transaction fees are present regardless of whether transactions are processed using traditional payment processors or they are done via the blockchain. However, it is important to understand if there are any benefits regarding transaction fees using the latter. For this reason, this part is dedicated to analyse and compare the two methods mentioned previously.

- **Credit card processing fees:** The specific fees charged by credit card processors can vary widely depending on the processor and the specific transaction. However, according to NerdWallet (2021), the average credit card processing fee is around 2.6%, with additional fees for things like chargebacks or foreign currency transactions.
- **PayPal fees:** For domestic transactions, PayPal charges a fee of 2.9% of the transaction amount plus a fixed fee of $0.30 per transaction, according to Paypal (2022). For international transactions, the fee is 5.0% of the transaction amount plus a fixed fee based on the currency being used.
- **Web3 approach:** Binance Smart Chain (BSC), which is the blockchain used by Binance's BNB token as well as the chain that is used in this project, charges a fee for each transaction. The fee is paid in BNB tokens and is used to motivate validators to process transactions on the blockchain. As of the date provided in the source, the current fee on BSC, according to BscScan (2022) is around $0.00001856 per transaction. However, it is important to note that other chains might have higher/lower gas fees, and smart contracts might have their own fees for using a certain protocol.

After comparing credit card, PayPal, and BSC chain fees, we can clearly see that the fees are much lower using the Web3 approach rather than using the Web2 approach for payments. According to this analysis, I am confident in saying that Web3 is financially more beneficial for a user than the traditional model seen in the majority of the Internet.

### 4.2.2 Data security comparison

In addition to fee comparison, I have also concluded a small research dedicated to analyse the safety of user data between Web2 and Web3 models. As mentioned in the 4.2.1 section, the Web2 model requires users to share sensitive financial information, such as credit card numbers, which may put users at risk of fraud or identity theft.

Airbnb and Booking.com data leaks:

- In 2023, a mysterious leak of booking data from the popular travel website Booking.com was reported. The leaked data included customer names, email addresses, and reservation details, and it appeared to have been accessed by a third party without authorization. The leaked data was subsequently used by scammers to send fraudulent emails to Booking.com customers, attempting to trick them into providing personal information or making unauthorized payments. Booking.com has advised customers to be vigilant about such scams and

has urged them to report any suspicious emails or activity. The incident highlights the ongoing threat of data breaches and the importance of companies taking steps to protect their customers' personal information. Source: Arstechnica (2023)

- In September 2020, an internal leak at Airbnb exposed some of the personal account information of its hosts. The leaked data included the hosts' names, email addresses, and phone numbers, as well as information about their bookings and costs. Airbnb immediately launched an investigation into the incident and notified affected hosts of the leak. The company emphasized that no guest information was compromised and that it has taken steps to prevent similar incidents in the future. Source: Computerweekly (2020)

In terms of the Web3 approach, decentralized blockchain networks such as Ethereum and Binance Smart Chain have many potential advantages in terms of data security. Transactions on these networks are recorded on public ledgers and verified by a network of nodes, making it harder for hackers to manipulate or change data. In addition, decentralized applications (dApps) built on these networks often use "smart contracts" to automate transactions and enforce rules without intermediaries such as banks or payment processors, further reducing the risk of data breaches. However, there have been some notable developments related to blockchain networks and dApps.

- The decentralized finance (DeFi) platform Poly Network was hacked in 2021, leading to the theft of more than $600 million in cryptocurrencies. However, the Poly Network team managed to quickly recover the stolen funds and return them to users. Source: CNBC (2021)

After the analysis, it is safe to say that while the Web3 approach has the potential to provide greater data security than traditional Web2 platforms, it is not immune to security risks and vulnerabilities. As with any technology, it is important for users to exercise caution and take appropriate security measures to protect their data and assets. However, it also shows the importance of companies taking steps to secure their data and protect the privacy of their customers, which could, in the near future, show a trend of companies dedicating resources to transition from the Web2 to the Web3 approach.

## 4.3 Authentication Process

With the help of Web3Auth and a custom backend, the authentication process is made more secure by not storing any sensitive credential data (such as passwords), with the only exception being email addresses, which will be hashed in future versions.

### 4.3.1 Web3Auth role

The project's user authentication, in addition to the backend JWT token signing and verifying, relies on Web3Auth, which is a decentralized authentication protocol that enables users to authenticate themselves with decentralized applications using their preferred blockchain wallet. Instead of relying on traditional username and password authentication, Web3Auth leverages the security of blockchain wallets to provide a more secure and user-friendly authentication method for dApps. When a user wants to access a dApp using Web3Auth, they simply connect their blockchain wallet to the dApp and sign a message to confirm their identity. In addition to just enabling users to authenticate themselves with their blockchain wallets via extensions such as MetaMask or Torus Wallet, Web3Auth also offers a social login feature that allows users to authenticate themselves

using their social media accounts, such as Google, Twitter, or Facebook. This feature is intended to make the authentication process more convenient for users who prefer to use their social media accounts for authentication. Upon login with a preferred social media account, a Torus wallet is created using Web3Auths Openlogin.

In order to understand the flow better, the high-level architectural diagram below shows how it works. Source: Web3Auth (2023)

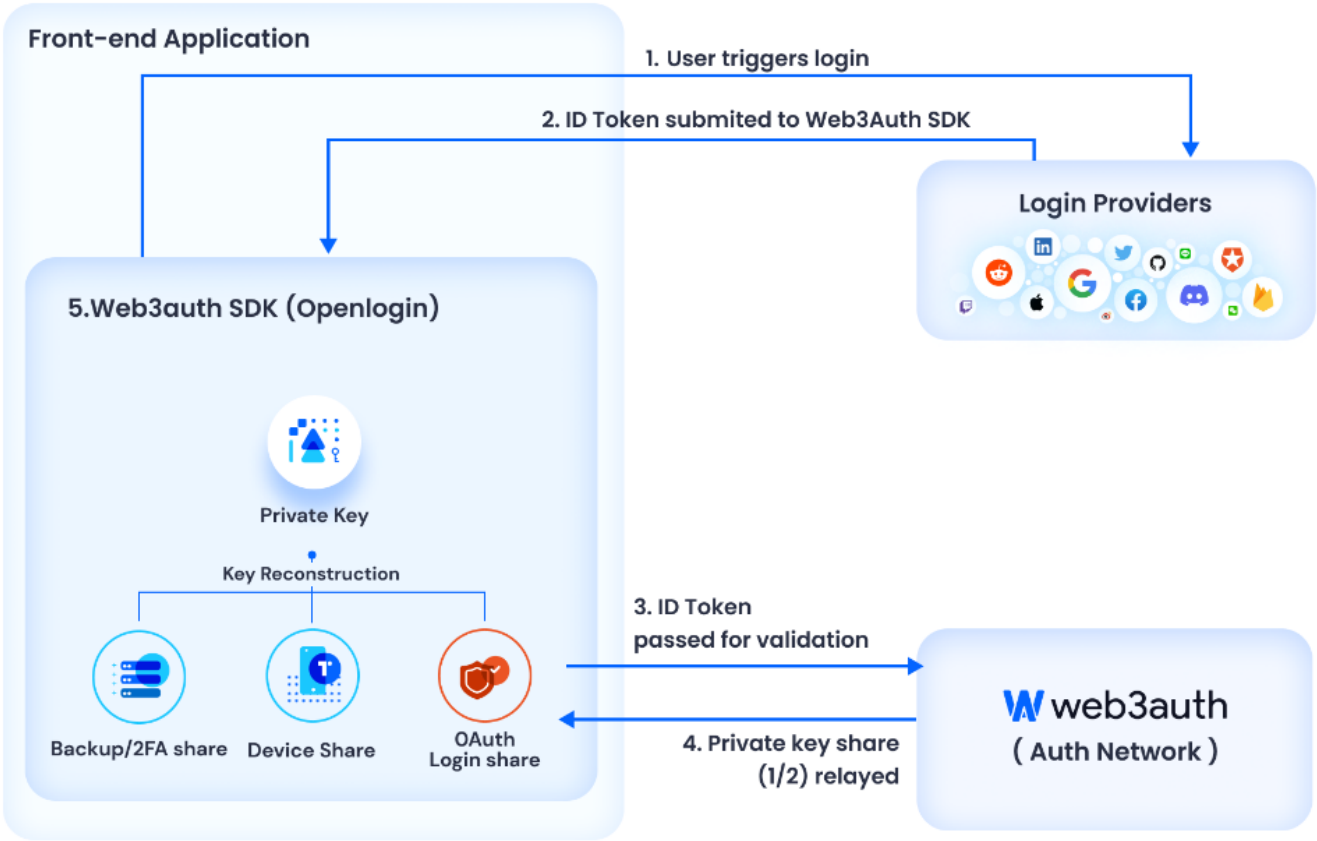


Figure 1. Web3Auth SDK flow.
**Source:** https://web3auth.io/docs/how-web3auth-works

### 4.3.2 Backend role

After the user is authenticated with Web3Auth, a JWT token is generated using the public ETH wallet address, the user's ID entry in the database the "users" table is hashed together with a secret seed, using the SHA-256 hash algorithm (see Figure 2 for better understanding).

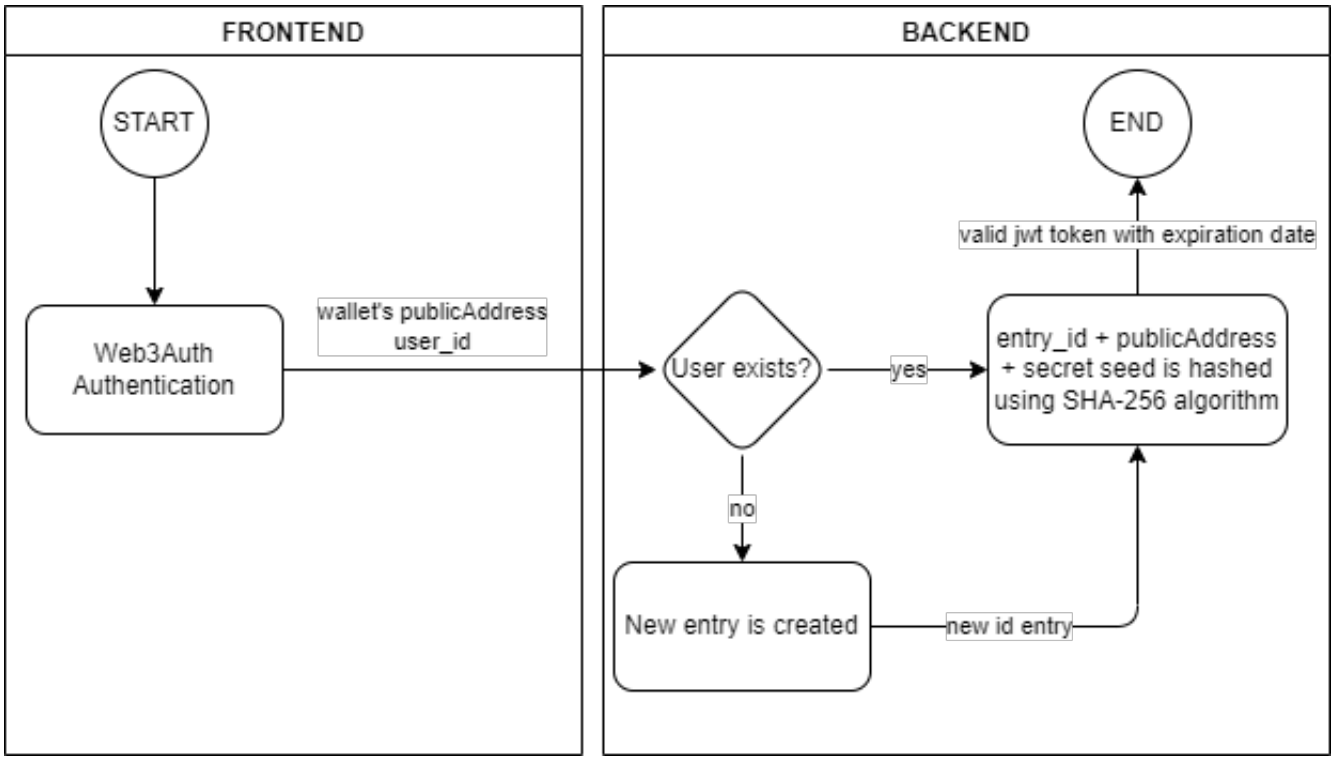


Figure 2. Authentication process

Upon successfully generating a JWT token, it is stored in the frontend using the Redux store and is accessed globally in JSX components using a custom selector hook. The data stored in the frontend is not mutable, however, in order to prevent users from modifying the request data and trying to reach protected routes that require JWT tokens, the backend has two middlewares before returning any data. See Figure 3 for visual representation.

- First middleware uses a custom express-validator function that checks the authorization header for the JWT token, and any data in the body that is expected (such as user_id).
- Second middleware verifies the token's integrity - checks the expiration time, then if the user's id that sent the request and the provided JWT's encoded id matches.

Passing these two middleware, any other route-dependant checks are run and the response is processed. The entire process is visualized in the diagram below.

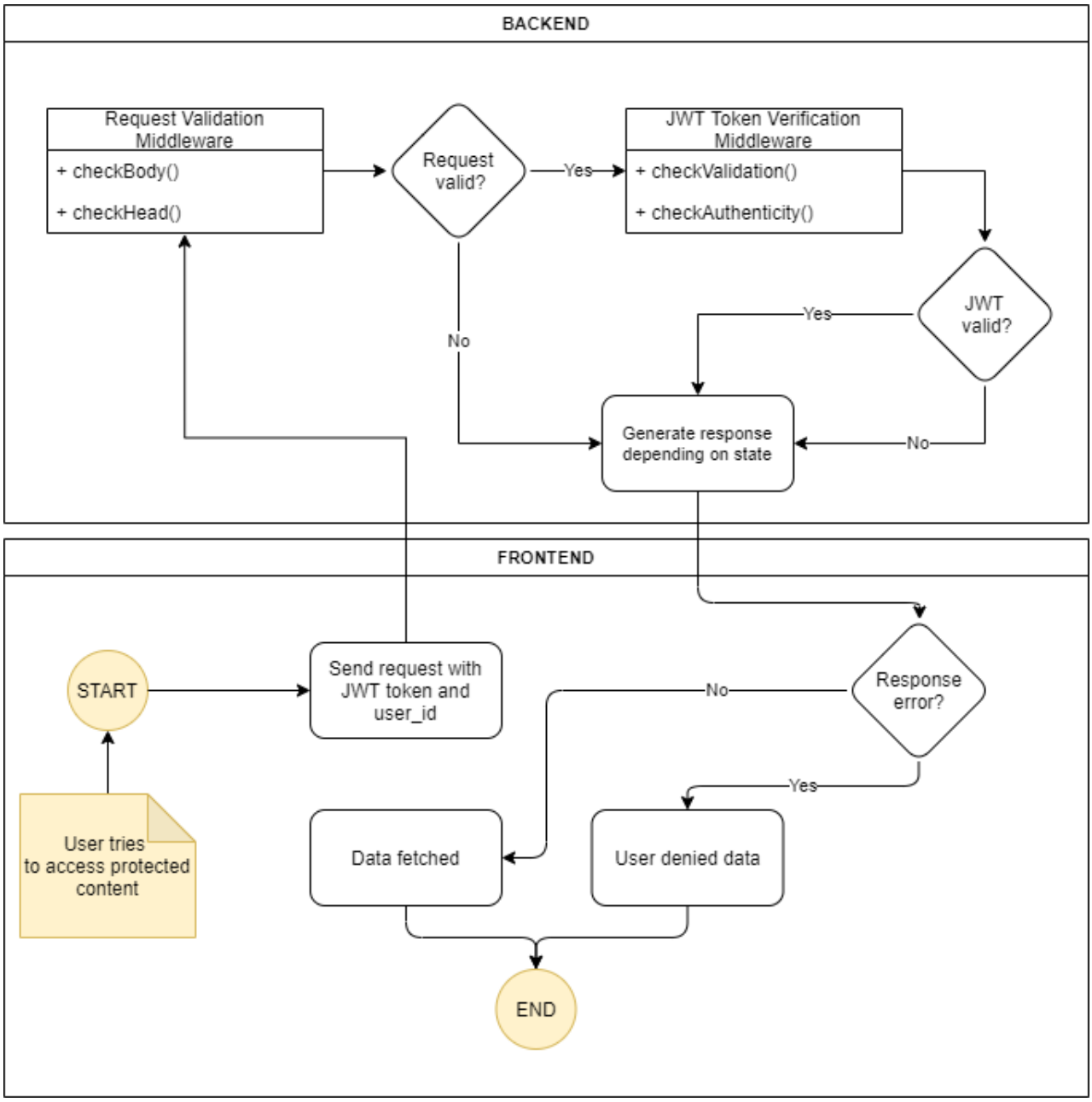


Figure 3. Protected route access Process

## 4.4 Payments

The payment process on this project is fairly complicated as it involves all parts of the system: the frontend to allow the user to interact with smart contracts and make payments, smart contracts to process and ensure secure fund transfers, and the backend to update reservation status and save transaction information; hence, it requires its own dedicated section that goes in-depth about the process. This process involves three parties: the user (the user that is making the transaction), the host (the user that owns the property), and the system owner (the person that deployed the payment contract); these parties will be called the user, host, and vault, respectively.

Upon making a reservation on a listing, the user has a short time frame to finish the payment for the reservation before it expires. First and foremost, the user is prompted to fill in the missing

contact information. This data will be used to contact the user if an issue or any other inquiry arises regarding the reservation. Secondly, the user has to approve a spending allowance before being able to interact with the payment contract. The amount the user has to approve is the same total amount the user will be paying for his stay, which is calculated by the following formula:

$$\mathbf{TotalAmount} = Nights * PricePerNight + ServiceFee$$

Spending allowance approval is done via interaction with the selected ERC20 token contract (currently USDT, which is a stable token whose currency rate to a dollar is 1:1, but this can be easily changed). If the transaction goes through, the user is prompted to complete the last part of the smart contract interaction, which is the payment itself. Since the payment contract can not transfer funds between parties if the spending allowance is not set, it is very important that the previous step be successful. That's why this step is only accessible to the user after the approval transaction goes through. If the user is using a wallet provider such as Metamask or Torus Wallet, they are provided with the option to input an amount they are willing to pay; otherwise, using social wallets, the user does not require any additional interactions, and everything should go smoothly. Upon input, the payment contract ensures that the amount specified is the correct amount and splits the funds according to the formula provided previously to the respective parties: the vault receives the service fee amount while the host receives the total amount minus the service fee. After the transaction is done, the user is informed about its status; considering it failed, the funds will not have been transferred, with detailed information of why it failed returned to the user, with the transaction hash included, which can be used to view on the Block Explorer website for any additional information needed. However, if the transaction went through and the funds have been transferred, the frontend calls backend to inform of the successful transaction. The backend then saves transaction information in the database and updates the reservation status to "Confirmed." At this point, the user receives a confirmation email with all the details of their reservation, including the payment information and reservation status. The system also sends a notification to the hotel staff to inform them of the new reservation and its status. The hotel staff can then prepare for the guest's arrival accordingly. In case there is an issue with any transaction in this process, such as insufficient funds or an invalid payment amount, the frontend will display an error message to the user and prompt them to try again. In order to understand this process better, refer to the process diagram below, which visualizes the process.

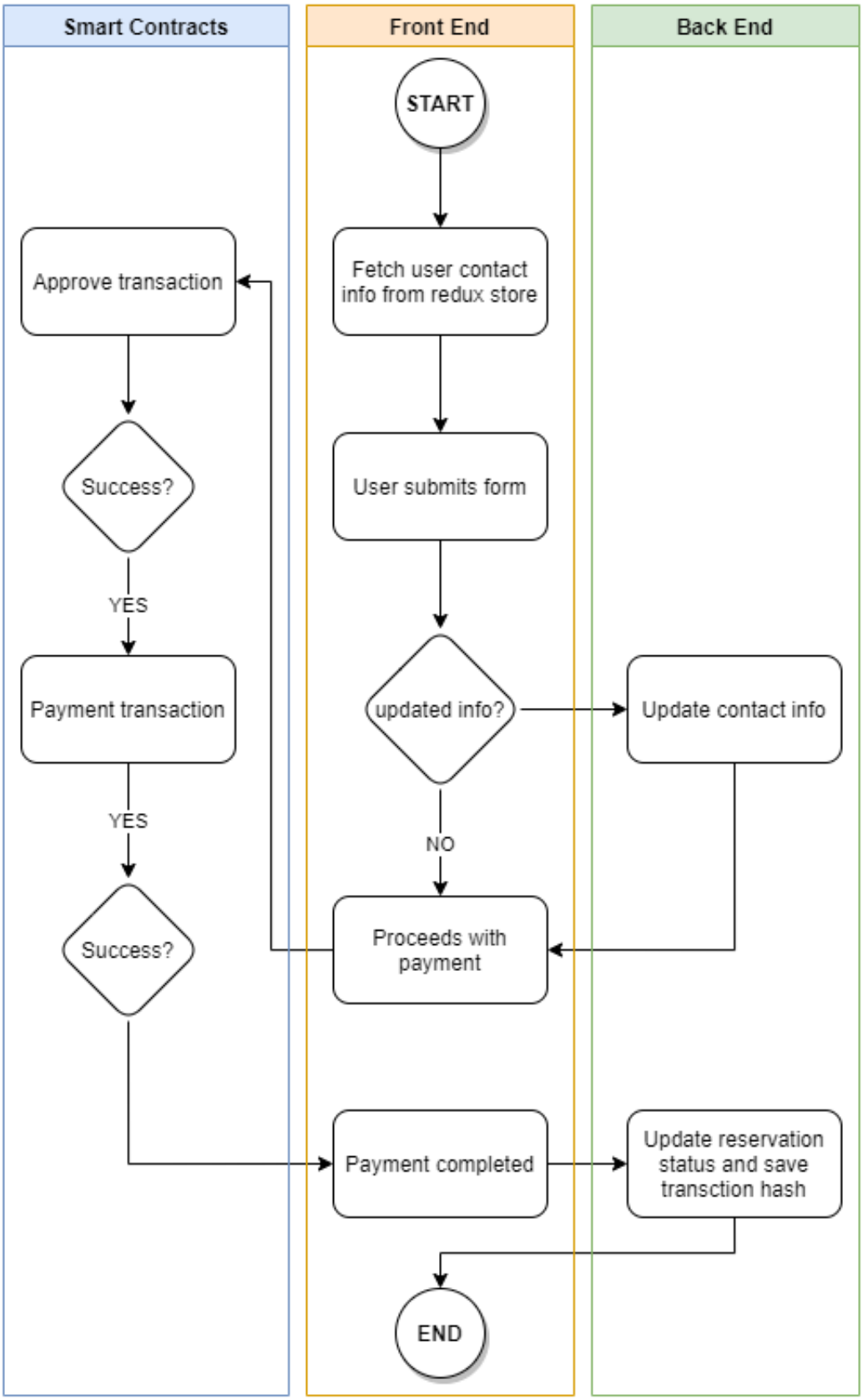


Figure 4. Payment Process

Overall, this booking system provides a seamless experience for both users and host staff, ensuring that reservations are made quickly and efficiently while minimizing errors and discrepancies and ensuring a safe, secure, and transparent way of transferring funds for services.

## 4.5 Payment streaming

In addition to the standard way of paying for bookings in the system, the user will be able to complete their payments in a period of time, by selecting "Pay as a Stream". Instead of paying the full amount instantly, the streaming method allows the user to finish paying for a booking in a set period of time.

The system utilizes a revolutionary streaming protocol called Superfluid, which enables real-time streaming of crypto assets from and to an account. Superfluid uses the Constant Flow Agreement (CFA). In a CFA, the sender agrees to have its account balance reduced at a certain per-second rate, which is called a "flow rate", and the receiving account's balance increase at that flow rate. The stream is perpetual, meaning that it can not be changed without modifying the agreement and will continue until the sender's Super Token balance hits zero. The Superfluid smart contract framework consists of few important components: Super Tokens and Super Agreements.

- **Super Tokens** - these are wrapped ERC-20 tokens that the user can use for engaging in a SuperAgreement, and may obtain by locking their ERC-20 tokens. These tokens do not have any liquid value, however, after unwrapping the tokens, the user gains liquid ERC-20 tokens that can be used in decentralized exchange systems to gain fiat value. Figure 5 visualizes the process of wrapping and unwrapping these tokens.

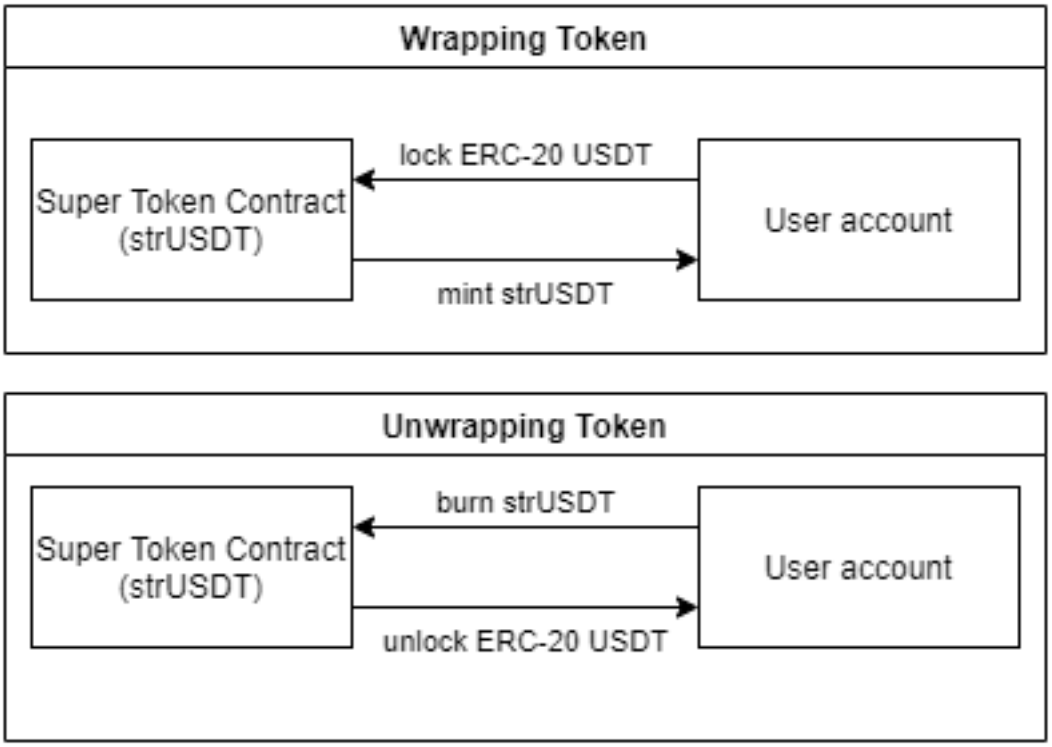


Figure 5. Super Tokens

- **Super Agreements** - is a way to empower and expand the capabilities of the Super Tokens. These agreements allow the constant flow streaming process. When an account engages in a Super Agreement with another account, it agrees to have its Super Token balance modified according to the rules of the Super Agreement and chosen parameters. In the case of the system, the user agrees to pay for their booking in a period of time, thereby creating a Super

Agreement between the host and itself, following that by the time Y the user will have paid X amount in a constant flow rate. This enables not only for the host to withdraw a fraction of payment whenever he desires, but also ensures that the user can manage their assets better.

This payment option is ideal for those who may not have the full amount upfront, but still want to secure their reservation. The system will deduct the selected currency from the user's account in real time, until the full balance is paid off. This not only provides flexibility for the user, but also ensures that the hosts receive payment in a timely manner. Furthermore, the streaming payment method is a convenient way to manage finances. Additionally, the system provides transparency and clarity throughout the payment process, so users can easily track their incoming and outgoing streams and stay on top of their finances.

# 5 Technologies

The technologies to create the system will be used as follows.

The MERN stack:

- Frontend: React.js + Typescript + Ethers
- Backend: Node.js + Express.js + Typescript
- Database: MySQL
- Smart Contracts: Solidity
- Design: CSS + MUI and styled-components libraries

**React.js** (Javascript library) is one of the most widely used libraries for developing smooth and reactive web applications worldwide. React uses a declarative approach to building UI components, making it easier to manage the complexity of applications enabling easy scalability. React also offers high performance, as it uses a virtual DOM that minimizes the number of updates required to render changes to the UI.

**Node.js** allows to use of JavaScript in the backend and is also widely used. It provides an event-driven, non-blocking I/O model that makes it ideal for building scalable and high-performance applications. Node.js also has a vast ecosystem of packages and libraries that make it easier to develop and deploy web applications.

**Express.js** is a popular web application framework built on top of Node.js. It provides developers with a minimal and flexible set of tools for building web applications and APIs. Express.js also supports middleware, which allows adding functionality to the system, such as authentication, error handling, and logging easier.

**MUI** is going to be used to save time by using the provided components for building a reactive, safe, and bug-free UI, while also using best HCI practices. In addition to these components, styled-components will be used to take the design even further and adjust it according to the needs.

**MySQL** is a popular, open-source database management system that enables efficient storage and retrieval of data.

**Solidity** is a high-level programming language designed for developing smart contracts on blockchain platforms like Ethereum. It enables to write secure and reliable code for decentralized applications (dApps) and smart contracts. Solidity's main purpose in this project is to develop the payment functionality used in the project. Additionally, the payments for the user will be more trustworthy, since the smart contracts are immutable and transparent because the code of the contract is open source.

To make it short, the MERN stack provides a powerful set of tools for building web applications that are scalable, efficient, and easy to maintain. React.js provides a robust and performant UI framework, Node.js and Express.js provide a flexible and scalable server-side architecture, and MySQL provides a reliable and scalable database solution. In addition, MERN main components use Javascript (to avoid errors and bugs, will be using Typescript instead) which I am familiar with.

# 6 Non-functional requirements

**SECURITY** The app will be using HTTPS requests between the backend and frontend to ensure data encryption. The system will be depending on OAuth provided by Web3Auth to handle sensitive user data such as login credentials, to combat the risk of a leak. No sensitive data will be stored on the database server. The user accounts will be protected by requiring the user to log in with chosen method (such as Social logins, Metamask, Torus wallet, or Walletconnect). Accessing protected routes to the backend will require a JWT authorization token, granted upon login with an expiration date.

**COMPATIBILITY** The website should be compatible with any popular browser such as, but not limited to - Firefox, Chrome, Microsoft Edge, and Opera GX. The website should be accessed by a machine running minimal hardware requirements.

**USABILITY** The website will apply best practices and other HCI principles to make the user experience easy and pleasant to use. The components that will be used to build the UI will be easily recognizable, because of the popular MUI library that many websites already use, with tweaking them by passing additional styling using styled-components library.

**PERFORMANCE** The website will not do any client-side calculations, and in addition to this, the website is going to be a single-page website using React.js and React router, that optimizes rendering based on the user's actions. This allows the application to perform well on any device running it. Moreover, the amount of backend calls will be reduced to a minimal amount by storing and caching data using Redux store, only updating it once an action has been performed that interacts with that data.

**RELIABILITY** The system must be reliable and available for use at all times, with minimal downtime or system failures that would otherwise prevent users from creating, searching, and accessing their reservations. Both the backend and frontend servers have fallback functions that handle any errors without crashing the server itself.

**RESPONSIVENESS** The web application should provide a responsive user interface, adapting to different screen sizes and devices. It should ensure optimal user experience on desktops, laptops, tablets, and mobile devices, allowing users to access and interact with the system seamlessly.

# 7 System architecture

## 7.1 Use cases

Figure 6 showcases the use cases of the system. There are three actors at play. The USER is a simple user that is able to use the majority of the functionalities that the system brings. The HOST is an advanced user, that inherits all of the simple USERS' use cases, in addition, has the ability to host his properties as available accommodation places. Finally, the ADMIN is a user that can manage users, properties, reservations, and view reports of regular users.

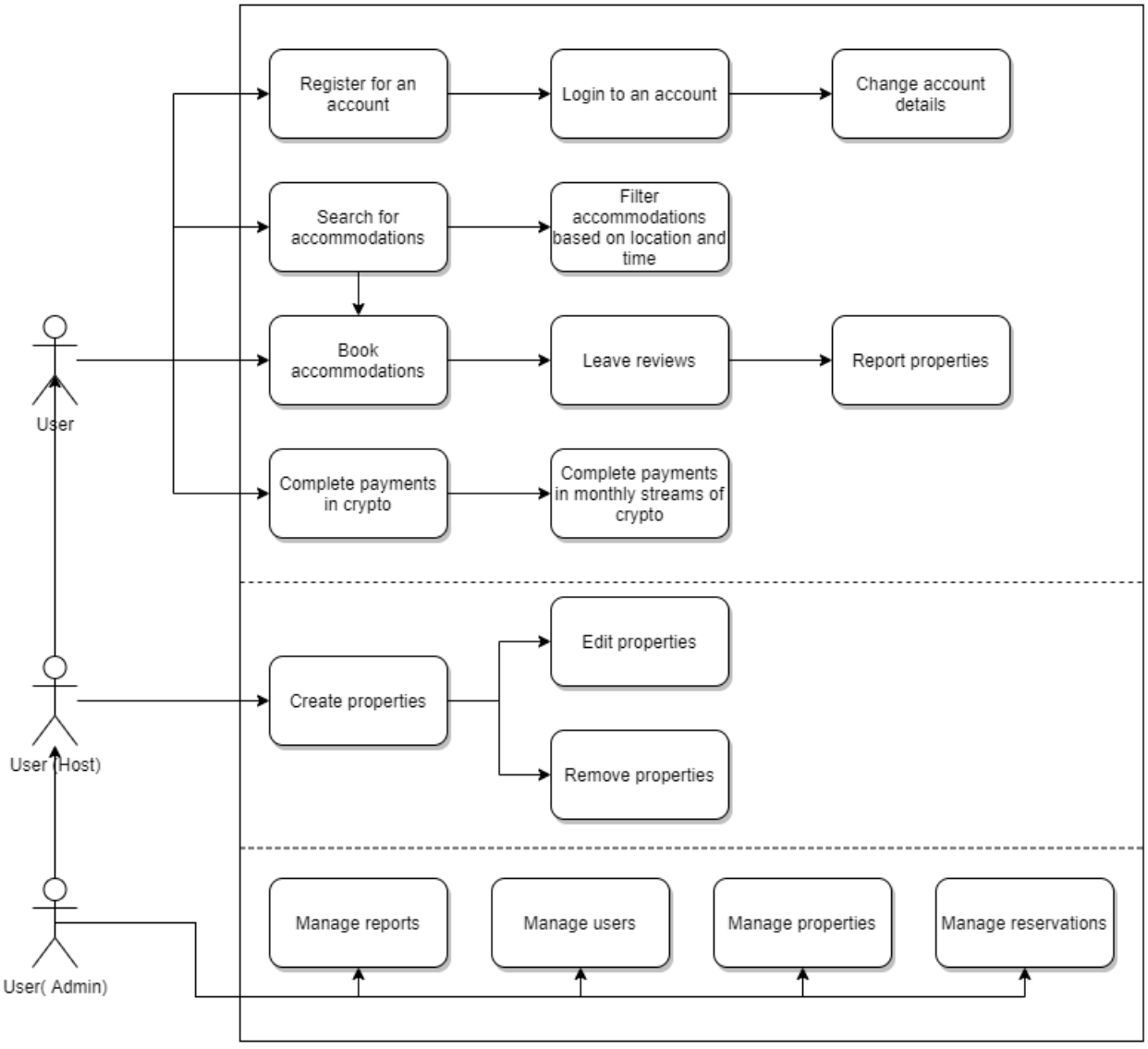


Figure 6. UML Use Case diagram

- **As a user I want to...**:
  - **Register for an account**: I can create a new account within the system with my ETH wallet. This allows me to become a registered user and access the system's features and functionalities.
  - **Login to my account**: Once registered, I can securely log in to my account using my ETH wallet. This grants me access to my personalized dashboard and settings,

providing a seamless user experience.

- **Change my account details**: I have the ability to modify my account information such as personal details, contact information, and preferences. This ensures that my account remains up to date and reflects any changes in my profile.
- **Search and filter accommodations**: I can easily search for accommodations based on my preferences, including location, and date range. This functionality allows me to explore a wide range of available properties that meet my specific requirements.
- **Book accommodations**: Once I find suitable accommodation, I can proceed with the booking process. This involves selecting my desired dates, providing the necessary contact information, and proceeding with the payment, ensuring a seamless booking experience.
- **Leave reviews**: After staying at a booked accommodation, I have the opportunity to leave reviews and ratings. This allows me to share my feedback and experiences with other users, assisting them in making informed decisions and fostering a community-driven environment.
- **Report properties**: If I encounter any issues or problems with a property, I can report it within the system. This functionality ensures that concerns related to properties are promptly addressed by the system administrators, maintaining the quality and integrity of the accommodations.
- **Complete payments via smart contracts in crypto**: I can make secure and transparent payments for my reservations using smart contracts in cryptocurrencies. This provides a seamless and efficient payment process, leveraging the advantages of blockchain technology. In addition, I want to be able to view my past reservations and payment history.
- **Complete payments via monthly streams of crypto**: For selected accommodations, I have the flexibility to complete payments using monthly streams of cryptocurrencies. This feature offers convenience and flexibility, allowing for real-time payments over the duration monthly period. In addition, I want to be able to view my incoming and outgoing streams.

- **As a host user I want to...(inherits cases from user)**
  - **Create properties**: As a host, I can create property listings within the system. This involves providing detailed information about the property, including descriptions, photos, amenities, and pricing. By creating accurate and attractive property listings, I can attract potential guests and maximize bookings.
  - **Edit properties**: I have the flexibility to edit the details of my existing property listings. This includes updating information such as availability, pricing, and amenities. By keeping my property listings up to date, I can ensure accurate and relevant information for potential guests.
  - **Delete properties**: In cases where I want to remove a property from the system, I have the ability to delete my property listing. This action removes the property from the system and makes it unavailable for future bookings. Hosts may choose to delete properties for various reasons, such as property unavailability or changes in ownership.

- **As a host user I want to...(inherits cases from user)**
  - **Manage reports**: I have the authority to review and manage reports submitted by users regarding properties. This includes assessing the validity of the reports, contacting users for further information if needed, and taking appropriate actions such as initiating investigations or resolving reported issues.
  - **Manage users**: As an admin, I have access to user management functionalities. This includes tasks such as user verification, ensuring compliance with system policies and guidelines, and taking actions such as suspending or deleting user accounts when necessary.
  - **Manage properties**: I have the ability to oversee and manage all properties within the system. This includes reviewing property listings, verifying their accuracy, and taking actions if necessary, such as flagging inappropriate listings or resolving disputes between users and hosts.
  - **Manage reservations**: Admins can view and manage reservations made by users. This includes handling booking-related inquiries, resolving conflicts or issues that may arise during a user's stay, and providing assistance to users when needed. Admins play a crucial role in ensuring a smooth and satisfactory experience for users throughout the reservation process.

## 7.2 Deployment

The system's deployment involves utilizing multiple components to facilitate its seamless operation and satisfy layered software architecture requirements. The system comprises four main parts, each serving a specific purpose. Figure 7 visualizes the deployment of the system.

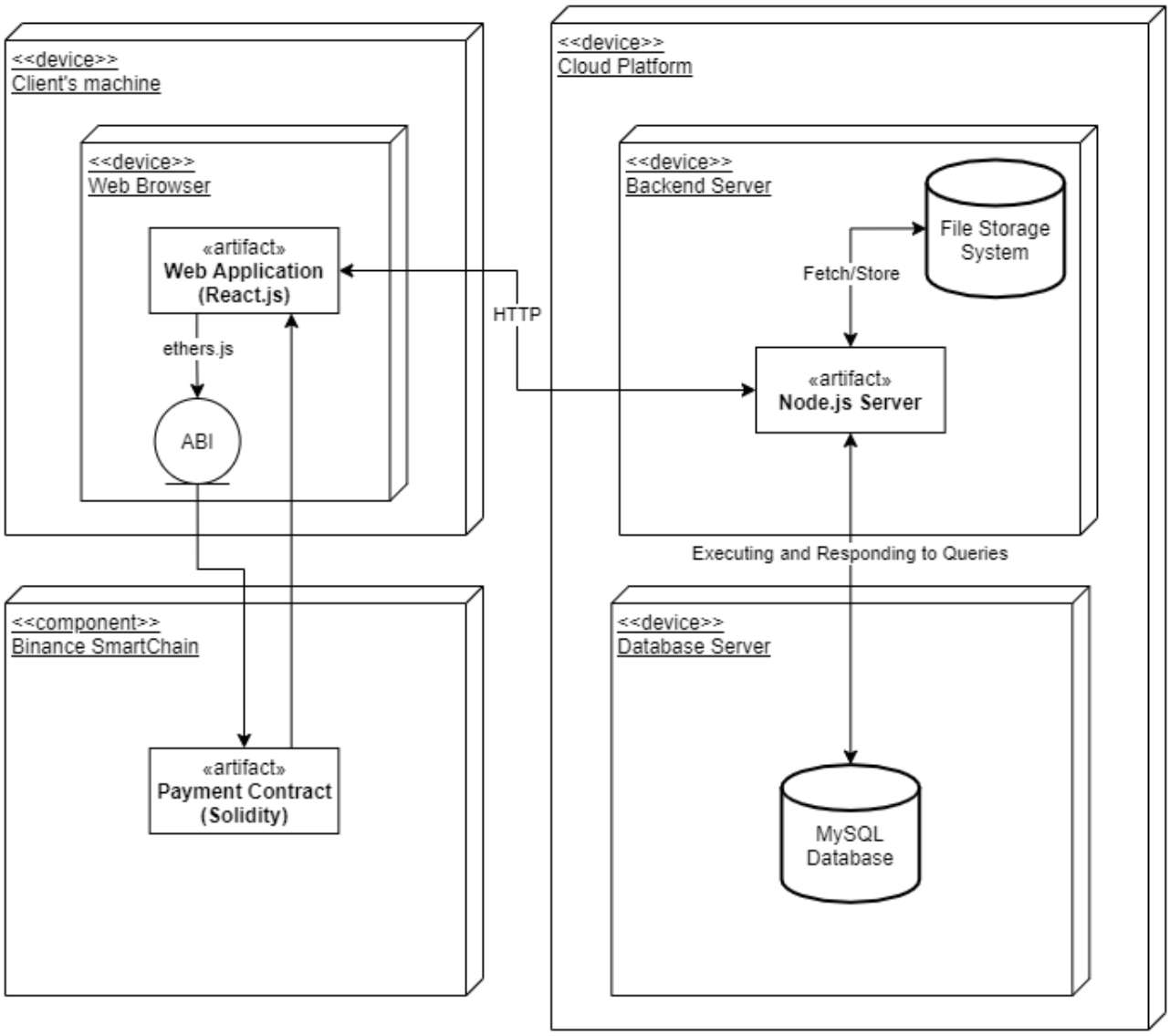


Figure 7. UML Deployment diagram

- **Client Machine**: The client machine acts as the interface for users to access the system. It runs the system's React.js web application, providing an interactive and user-friendly experience. Users can interact with the system by accessing the web application through their web browsers. The web application is responsible for the presentation layer.
- **Cloud Platform**: The cloud platform serves as the hosting infrastructure for the system, housing two supplementary servers: the backend server and the database server.
  - **Backend server**: It runs a Node.js server and assumes a crucial role in the architectural framework of the system. It acts as a bridge between the frontend (client) and the database server, providing seamless communication and data exchange. Additionally,

the Node.js server assumes responsibility for overseeing authentication and authorization processes, ensuring secure access to the system. In addition, it plays an essential role in the storage of files within the file storage system, by acting as a middleware and sanitizing user file input (e.g. property pictures). It acts as the service layer in the system.

  - **Database server**: The database server runs a MySQL RDBMS, that acts as centralized data storage location for various system data. It holds some essential information such as transactions, properties, and other relevant data required for the system's functionalities. This server ensures the persistent storage and retrieval of data, allowing for efficient data management within the system. It acts as the data layer in the system.

- **Binance SmartChain**: The system utilizes the Binance SmartChain, a blockchain network, to effectively underpin its operations. The Binance SmartChain provides the decentralized and crucial infrastructure for executing smart contracts, which are integral to specific functionalities within the system. Alongside the Payment Smart Contract, the Binance SmartChain also hosts contracts responsible for enabling crypto streaming functionality within the system. In addition, the system utilizes Ethereum wallets as user accounts, leveraging the security and flexibility provided by the Ethereum blockchain. Each user account is associated with a unique Ethereum wallet address, enabling secure and decentralized management of user identities and transactions.

## 7.3 Relational Model

In order to better understand how data will be stored in the MySQL database server, the following relational model, as well as a short description is provided. In addition, view table descriptions are also below the relational model schema.

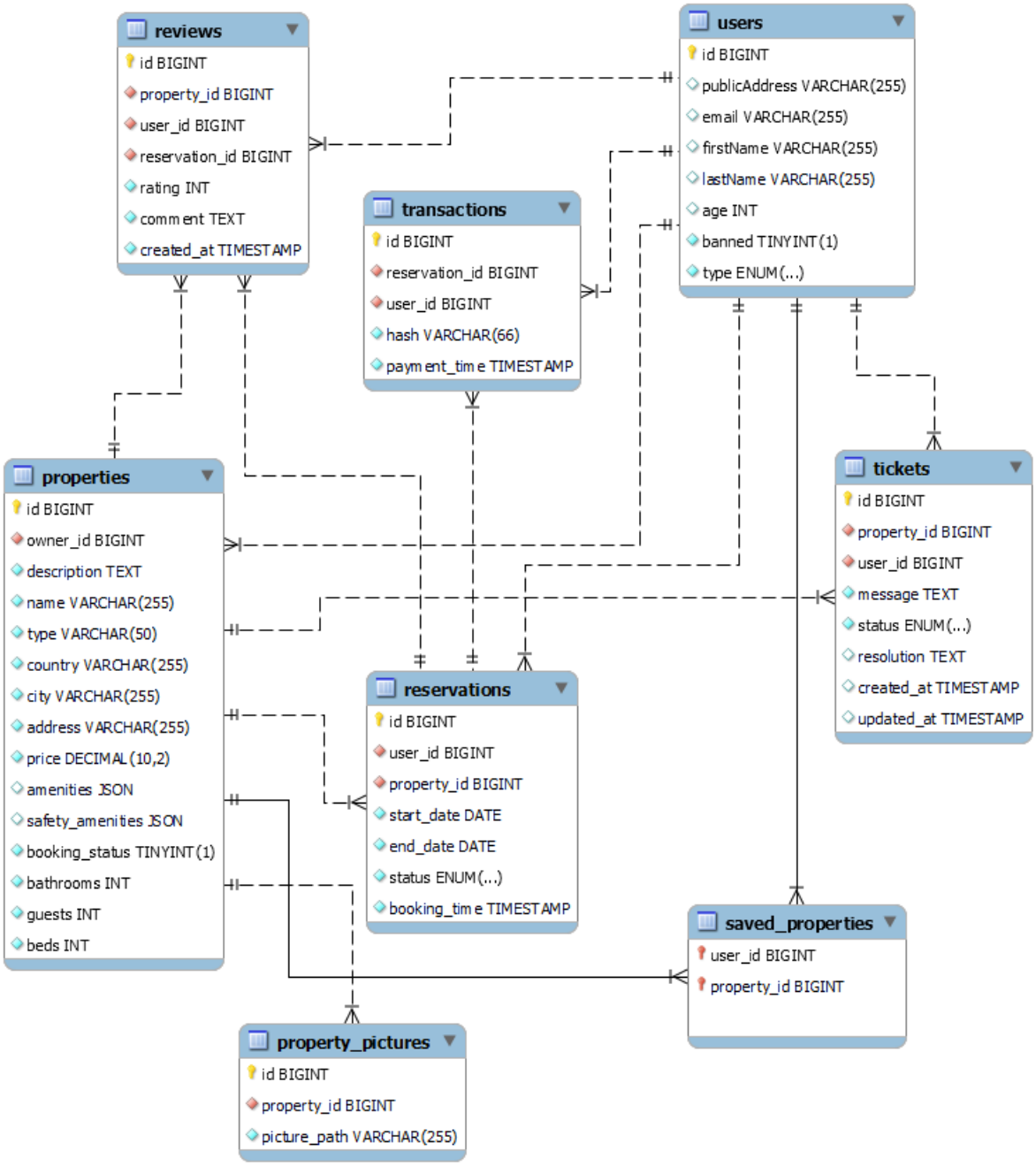


Figure 8. Database Relational model

**View tables:**

In addition, to reduce the complexity of the queries, we are using some view tables that joins multiple existing tables. View table **property_with_pictures** is joining property table with property_pictures, and is used to retrieve all the picture paths as a string separated by commas inside each property row.

**Tables:**

- users
  - id - a unique identifier
  - publicAddress - a public ETH address of the user
  - email - hashed email address of the user
  - firstName - the name of the user
  - lastName - the surname of the user
  - age - the age of the user
  - banned - boolean type if the user was banned
  - type - a type of the user (regular/host/admin)
- properties
  - id - property id
  - owner_id - property owner id
  - description - property description
  - name - property name
  - type - property type (eg. Condo, House)
  - country - country that the property is in
  - city - city that the property is in
  - address - property address
  - price - price of the property per night
  - amenities - property amenities (eg. Wifi)
  - safety_amenities - property safety amenities (eg. Fire alarm)
  - booking_status - boolean if the property is currently booked
  - bathrooms - property size information
  - guests - property size information
  - beds - property size information
- saved_properties
  - user_id - user that saved the property id
  - property_id - saved property id
- reservations
  - id - reservation id
  - user_id - user that reserved the property id
  - property_id - property id

  - start_date - start date of reservation
  - end_date - end date of reservation
  - status - reservation status with the value of (Confirmed, pending_payment, expired, canceled)
  - booking_time - indication when the reservation was made
- tickets
  - id - unique id of a report
  - user_id - id of the user that submitted the report
  - property_id - property that was reported
  - message - report message
  - status - status of the report with the value of (OPEN, RESOLVED, CLOSED)
  - resolution - admin message about the resolution.
  - created_at - date of the report creation
  - updated_at - date of the report update
- reviews
  - id - unique id of a review
  - property_id - property id that was reviewed
  - user_id - user id that reviewed the property
  - rating - rating (0-5)
  - comment - user left comments
  - created_at - date of the review
- property_pictures
  - id - unique id of the entity
  - property_id - id of the property that the picture belongs to
  - picture_path - the path of the picture in the file storage server.
- transactions
  - id - unique id of the entity
  - reservation_id - id of the reservation
  - user_id - id of the user that made the transaction
  - hash - transaction hash on the chain
  - payment_time - date of the transaction.

# 8 App Screenshots

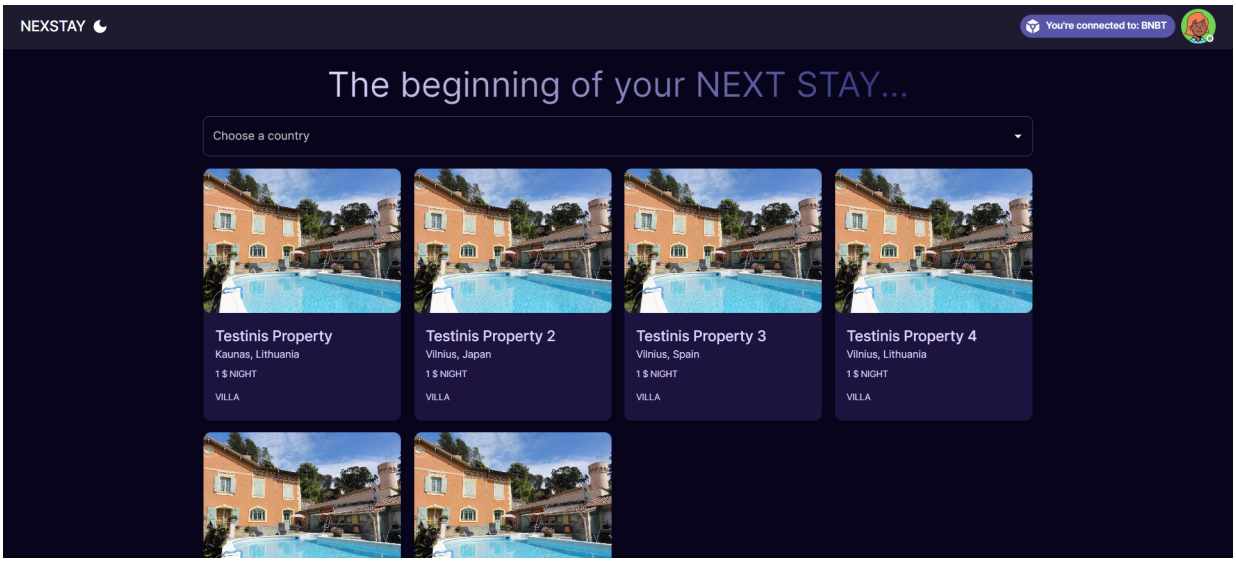


Figure 9. System's landing page

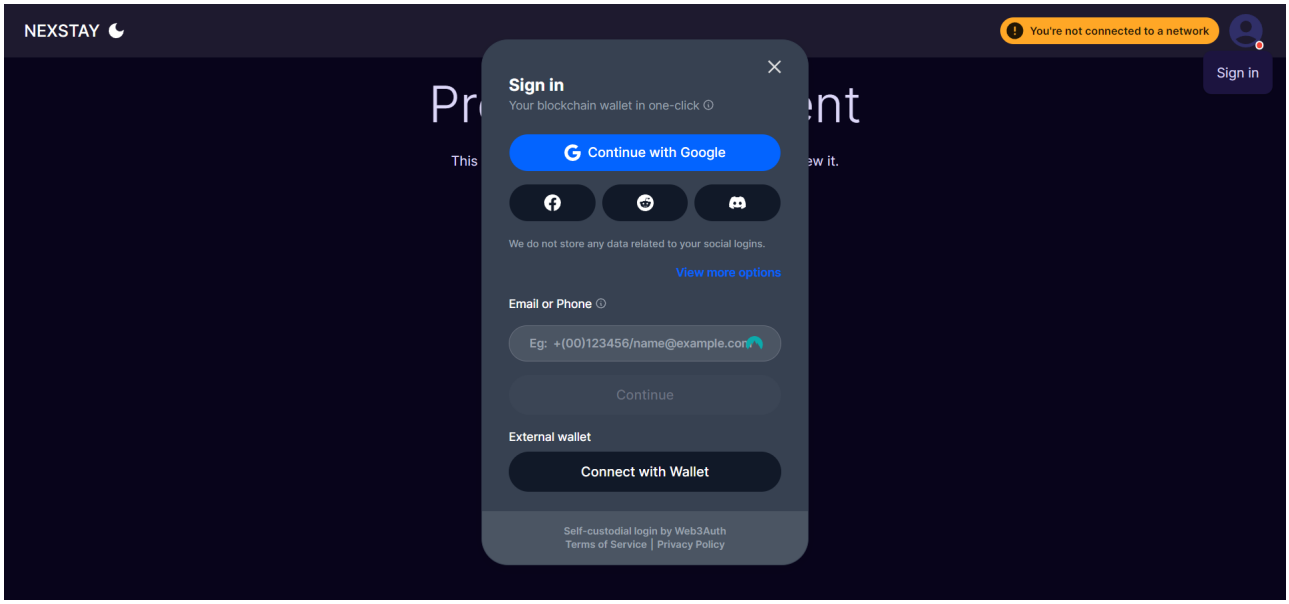


Figure 10. Login modal

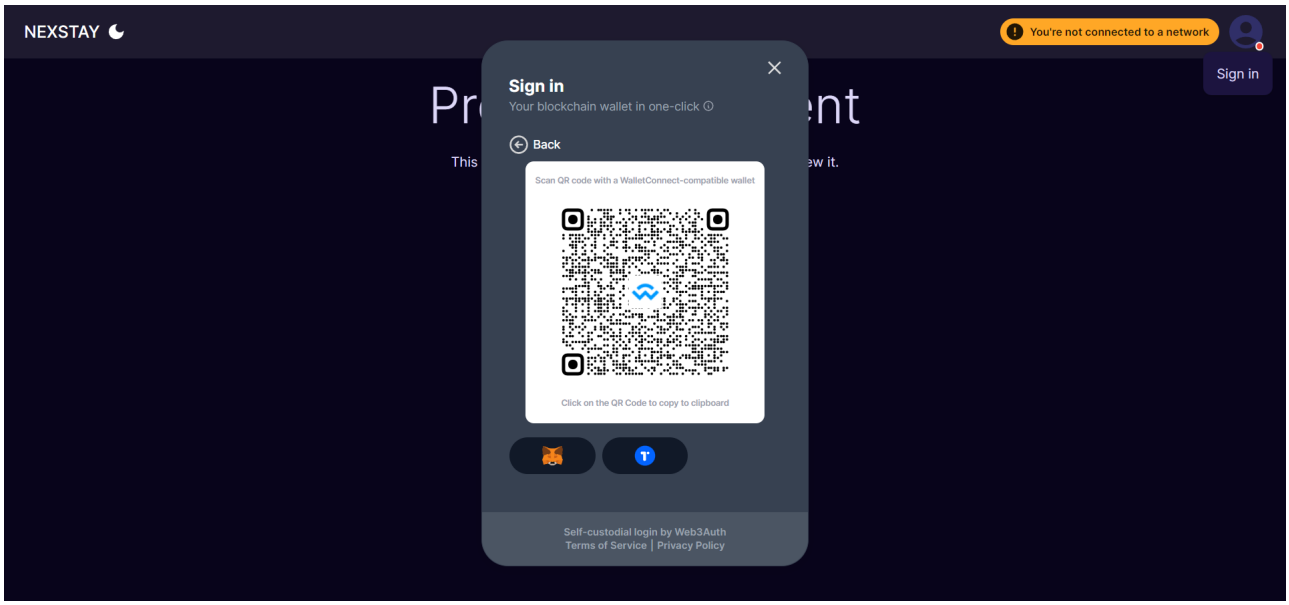


Figure 11. Login modal display wallets

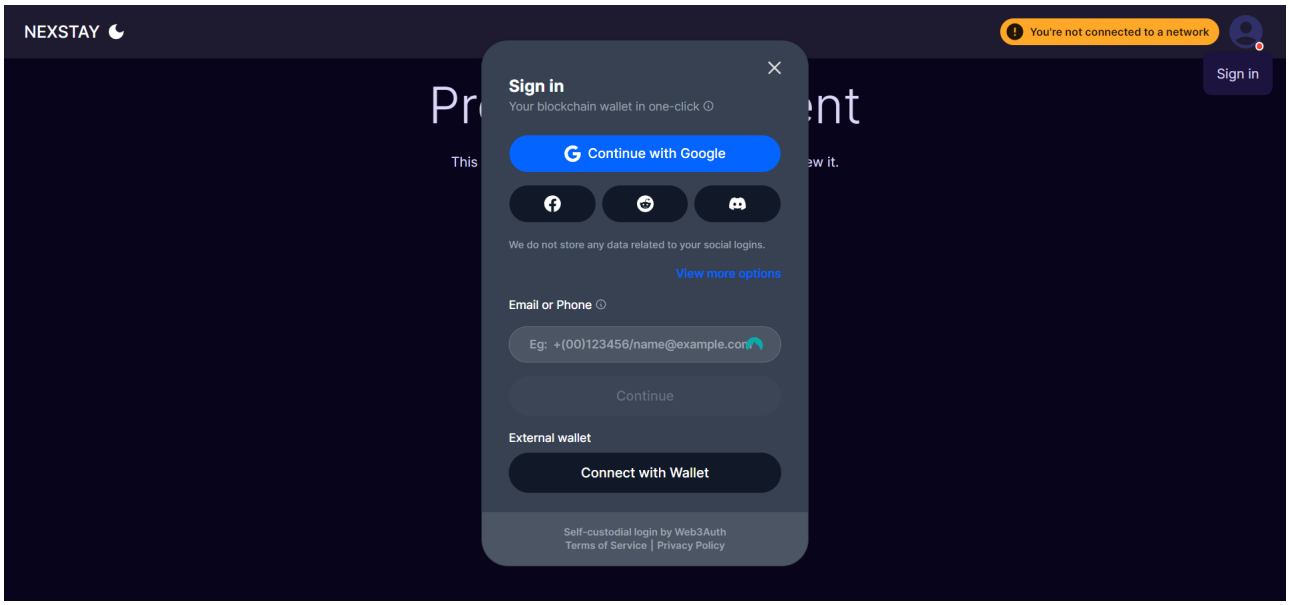


Figure 12. Login modal logging in

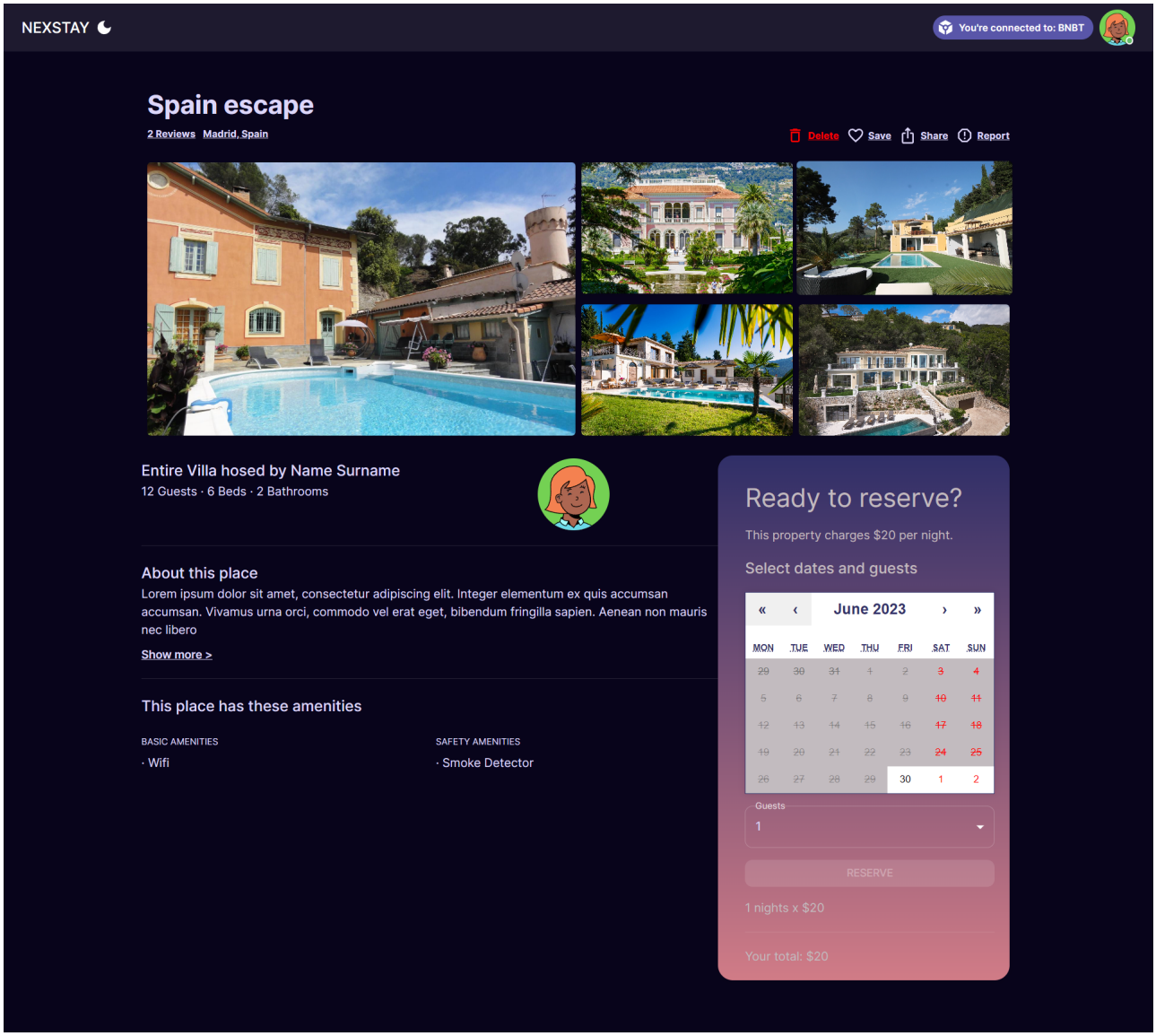


Figure 13. Property page

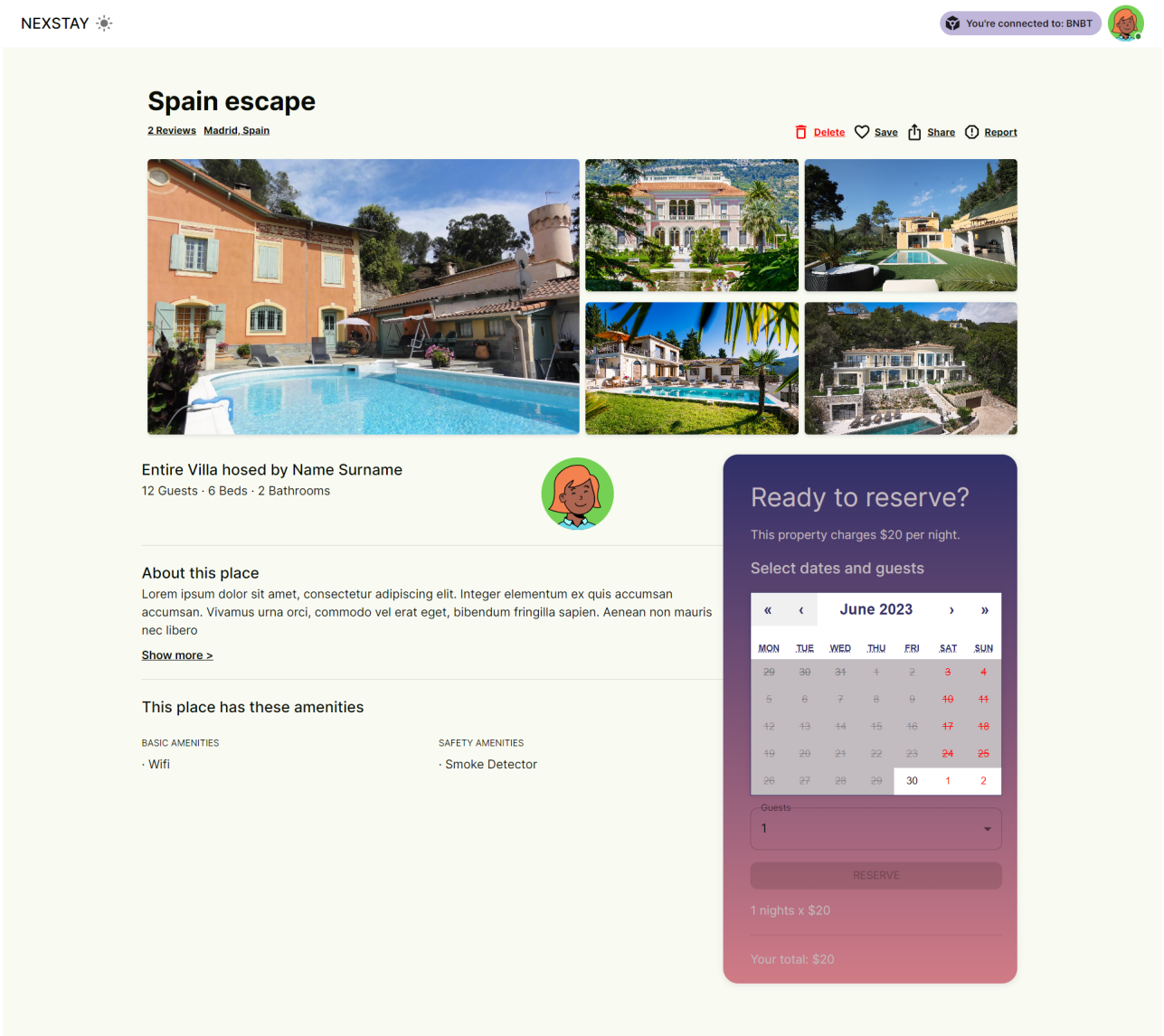


Figure 14. Property page (light theme)

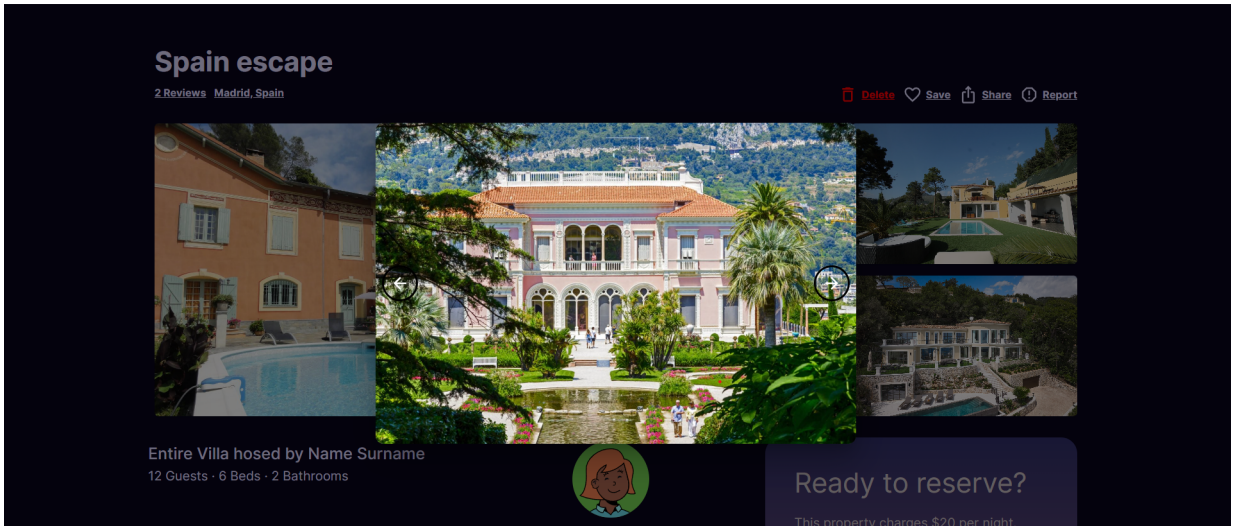


Figure 15. Property page picture view

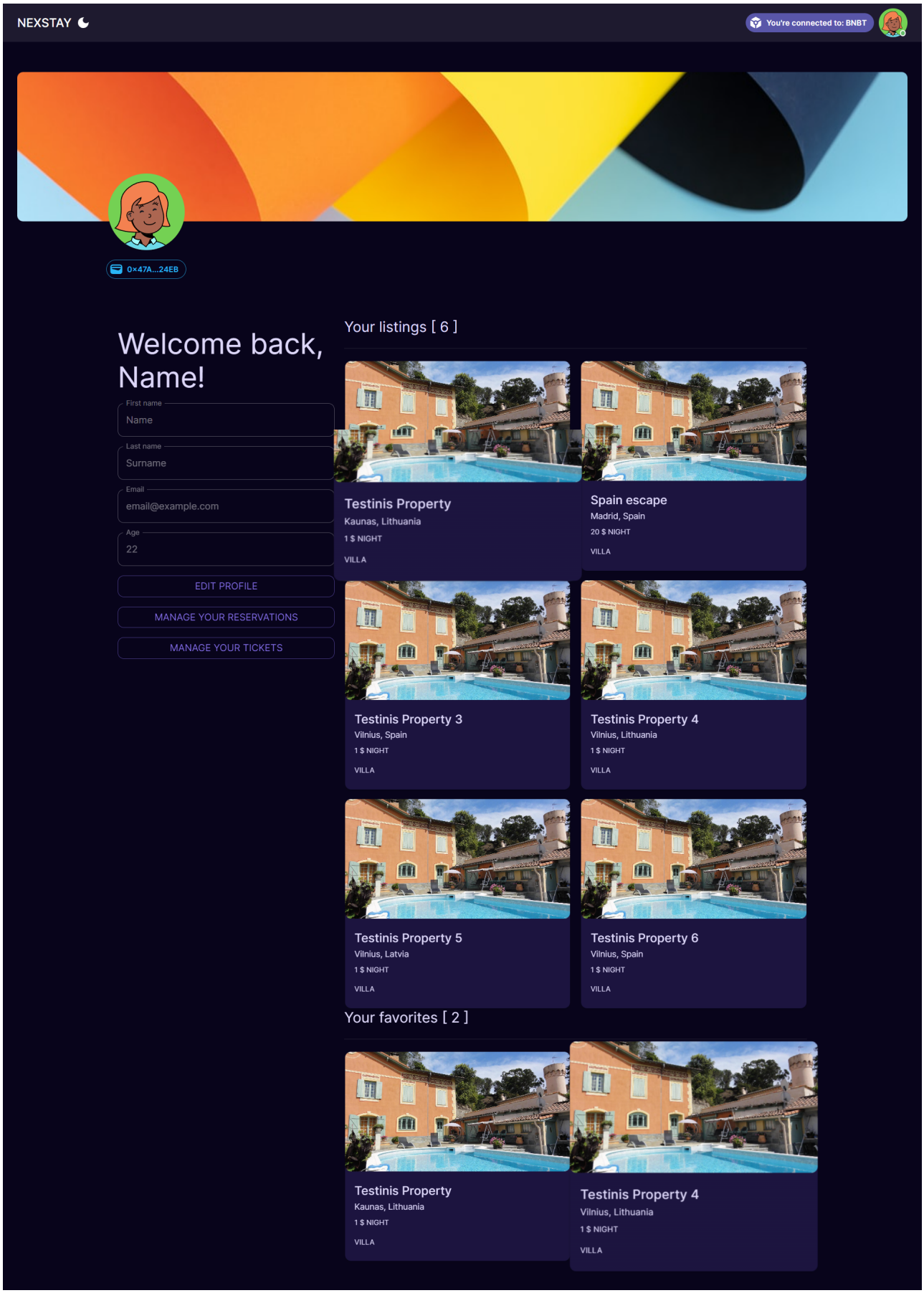


Figure 16. User profile page

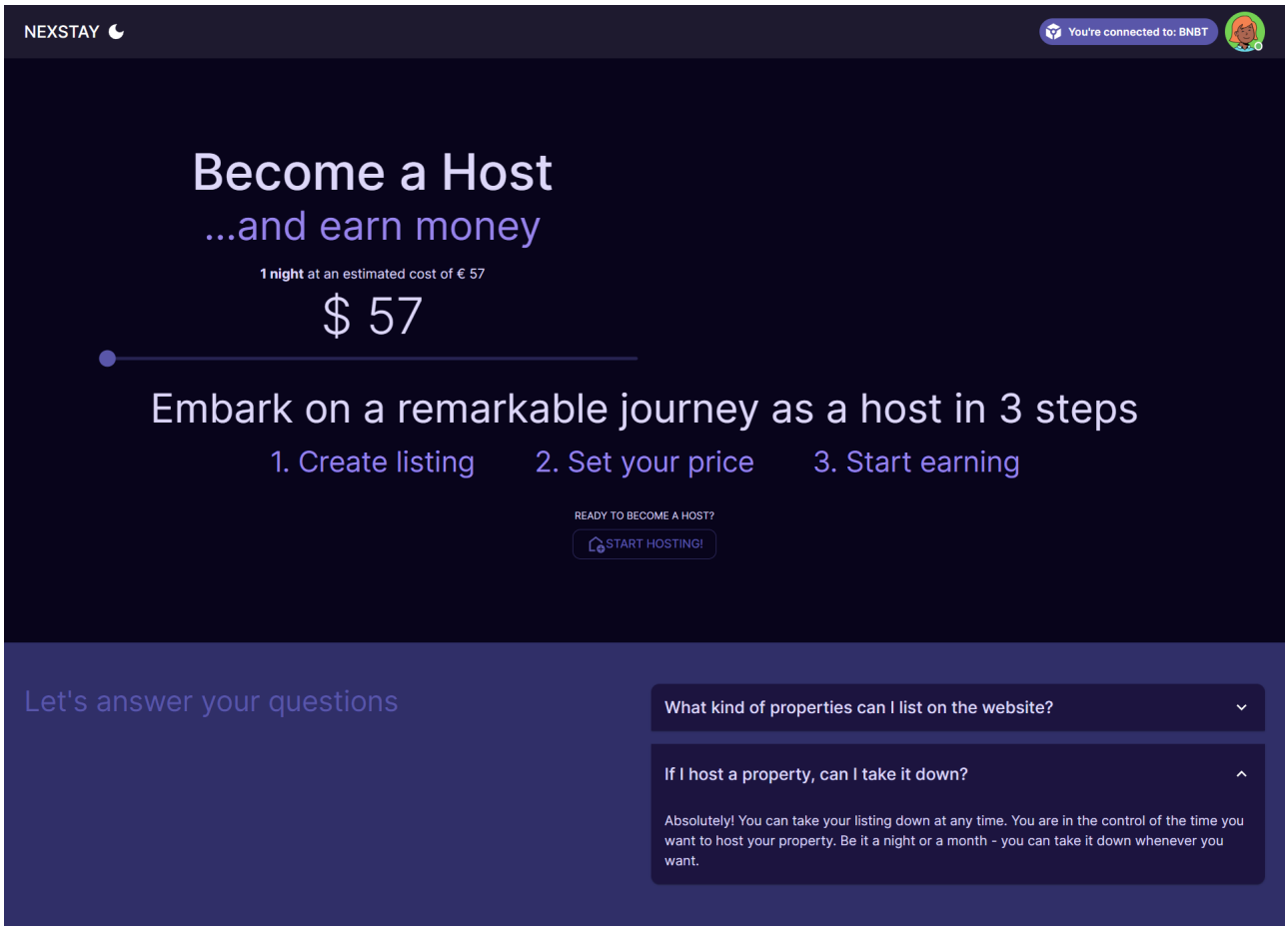


Figure 17. Host landing page

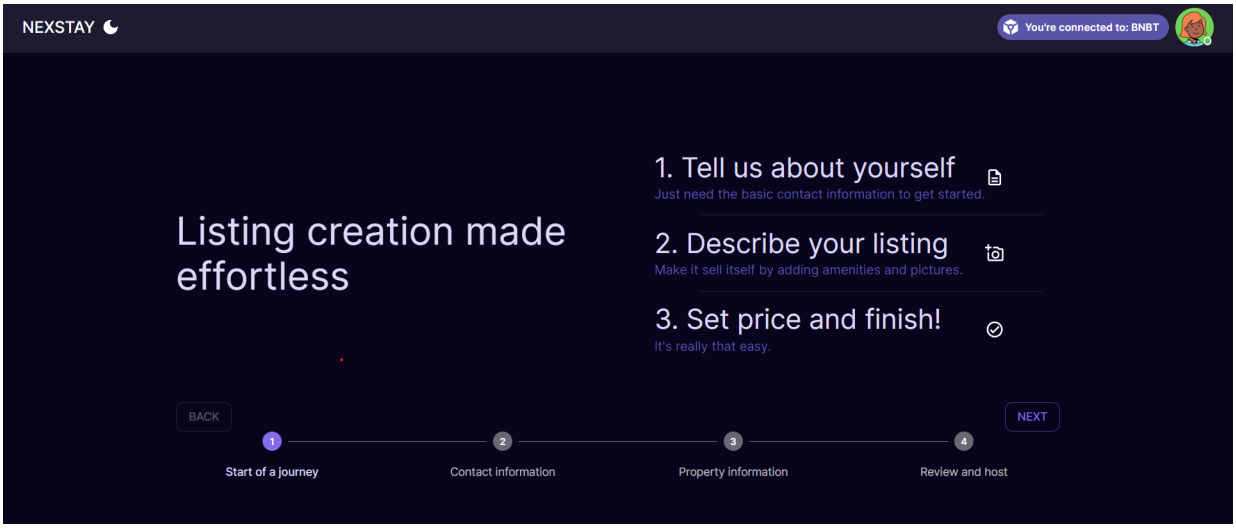


Figure 18. Property listing page

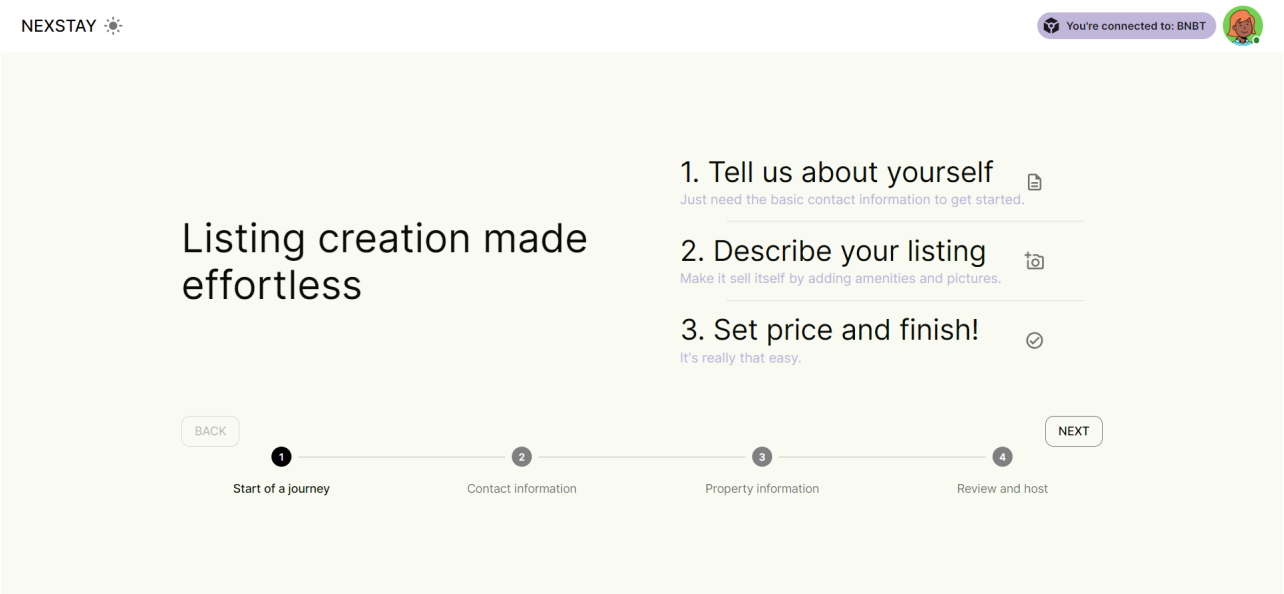


Figure 19. Property listing page (light theme)

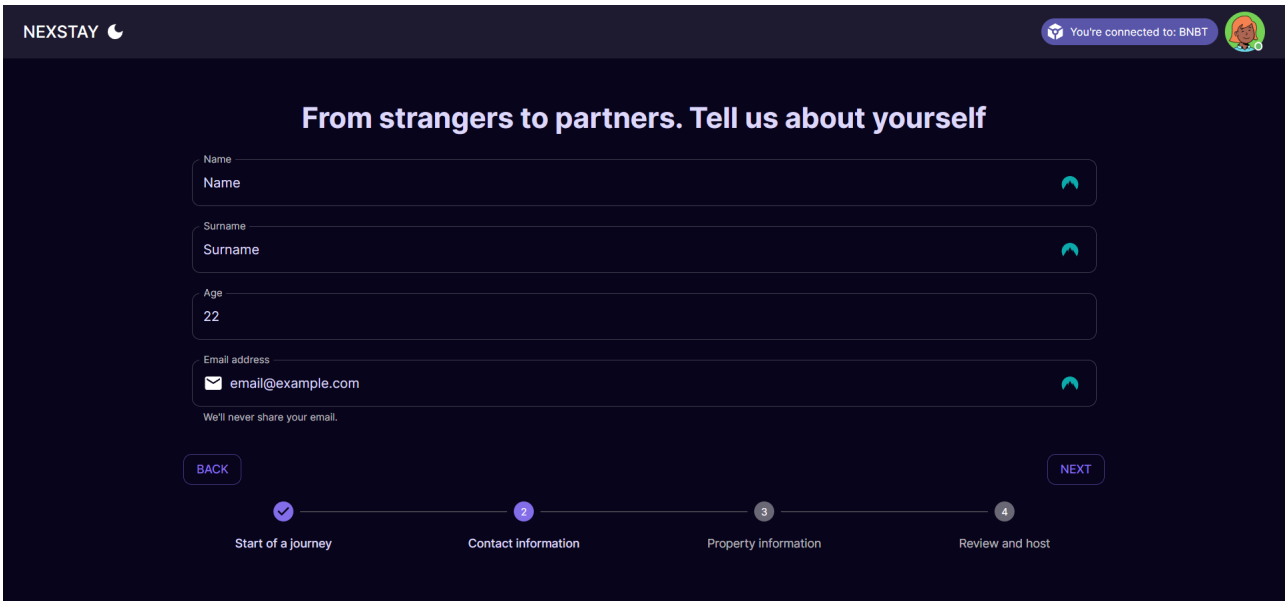


Figure 20. Add property info page

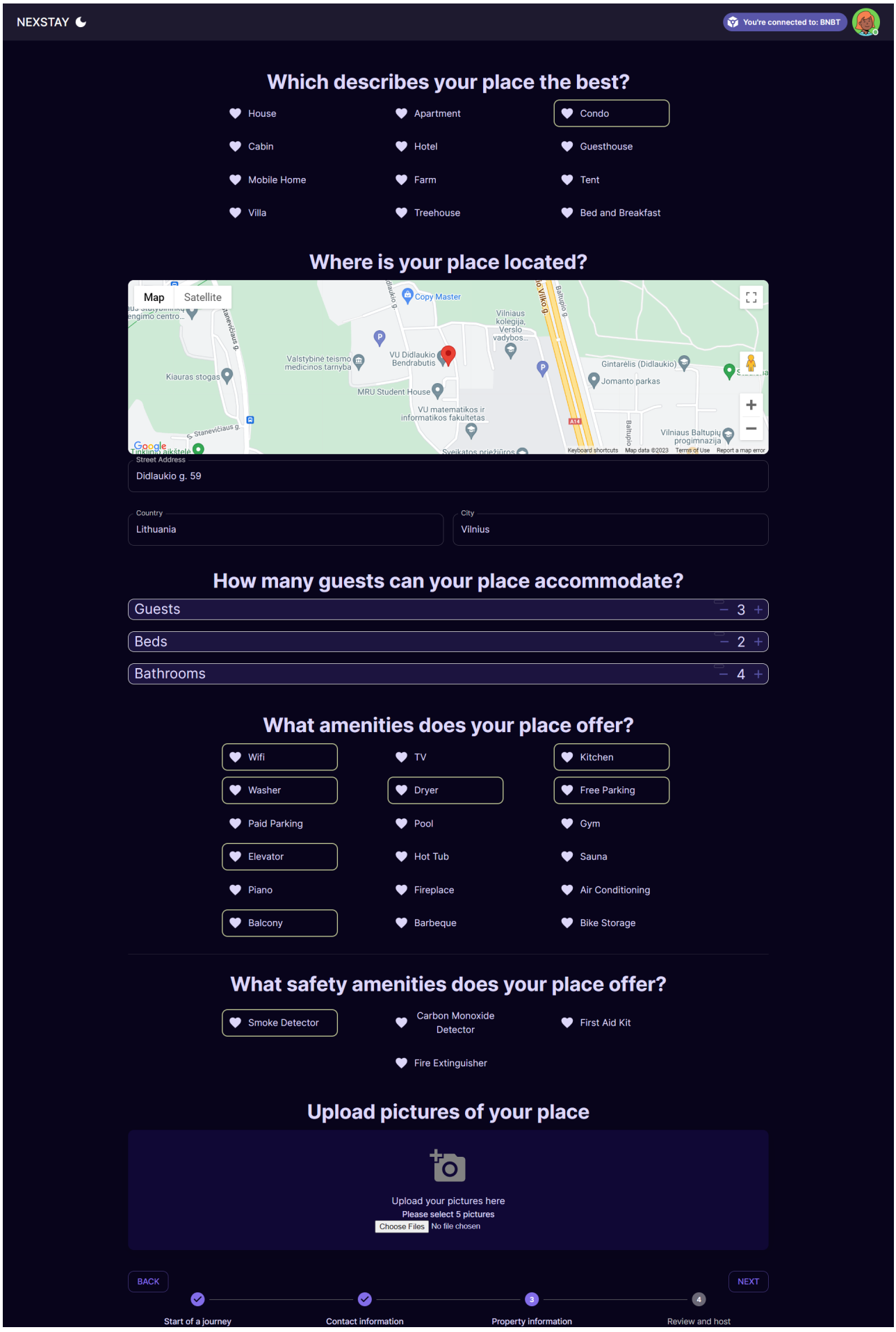


Figure 21. Add property info page 2

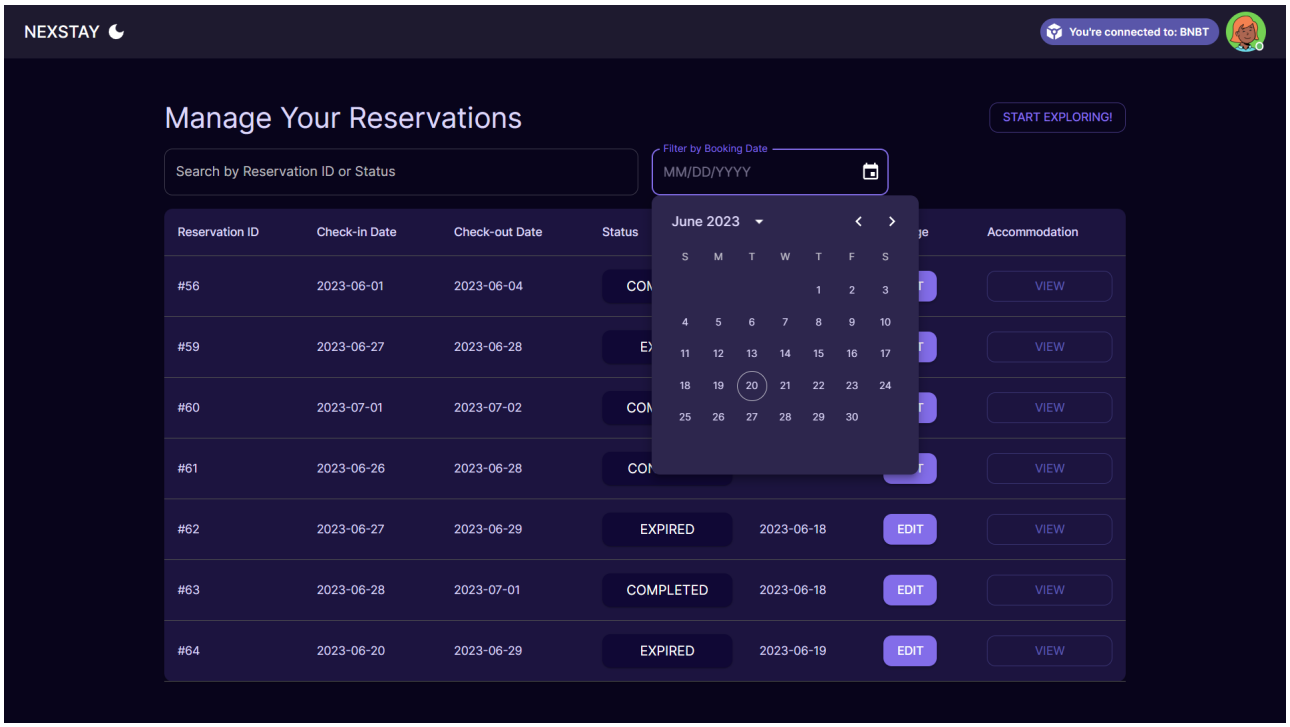


Figure 22. Reservation page

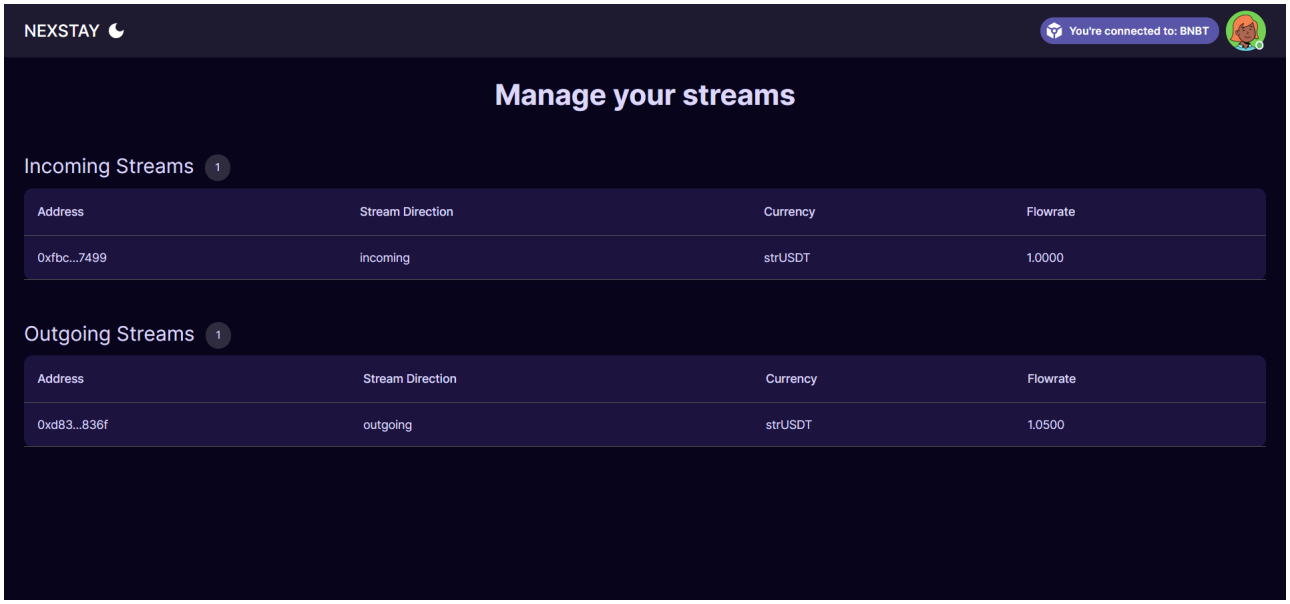


Figure 23. Streams page

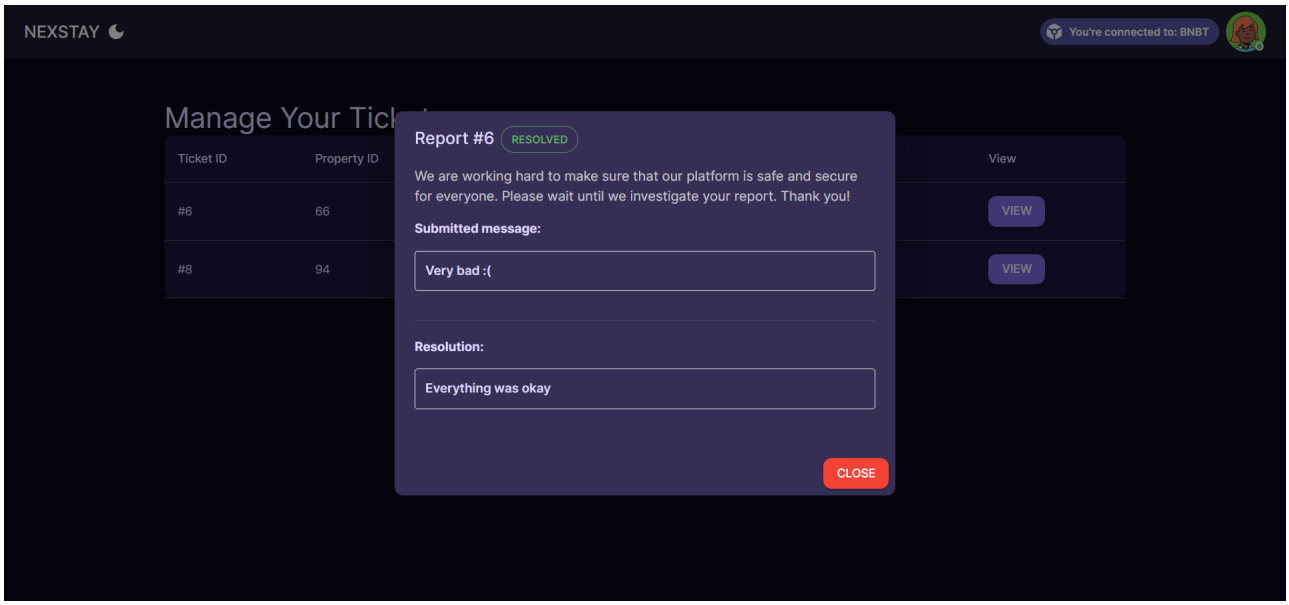


Figure 24. Tickets page

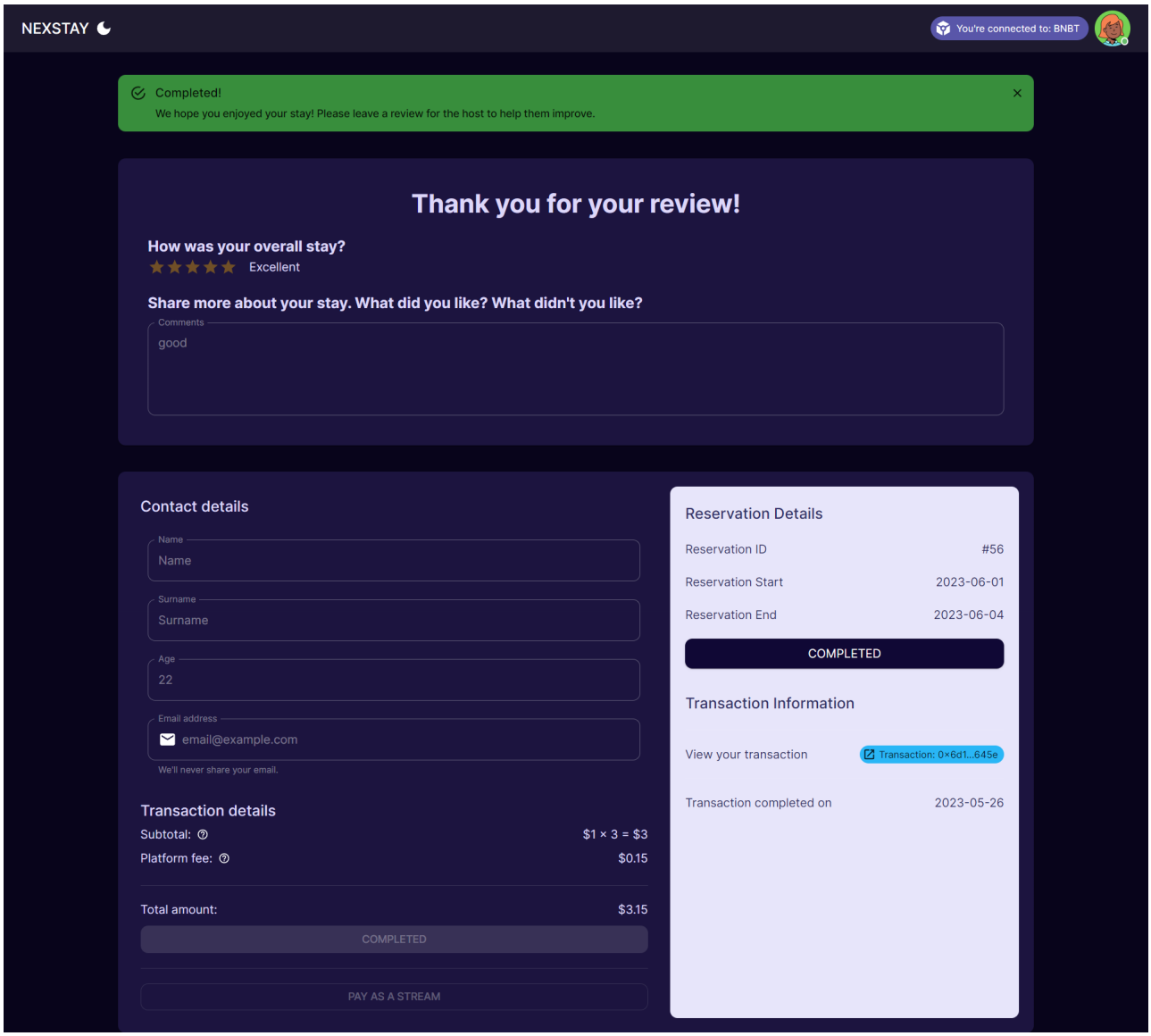


Figure 25. Completed booking

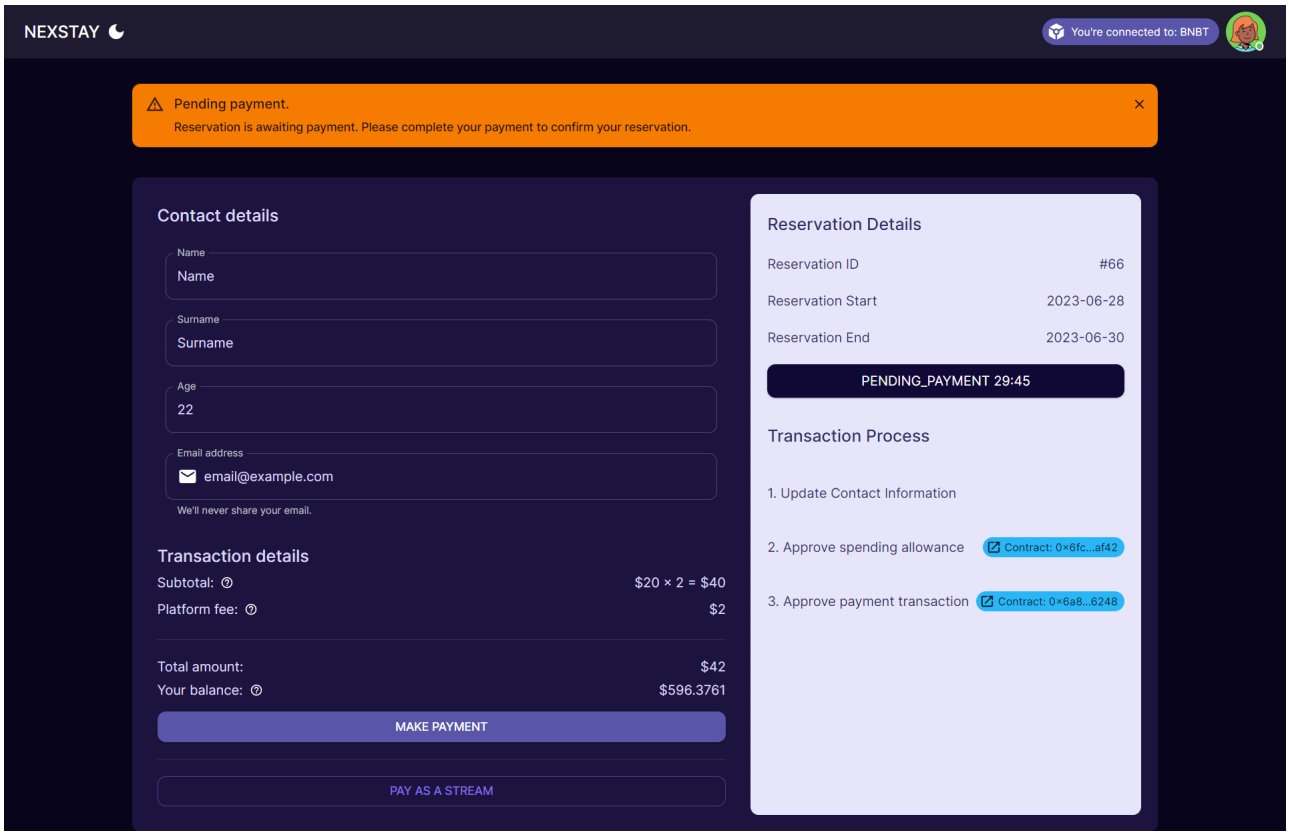


Figure 26. Pending payment page

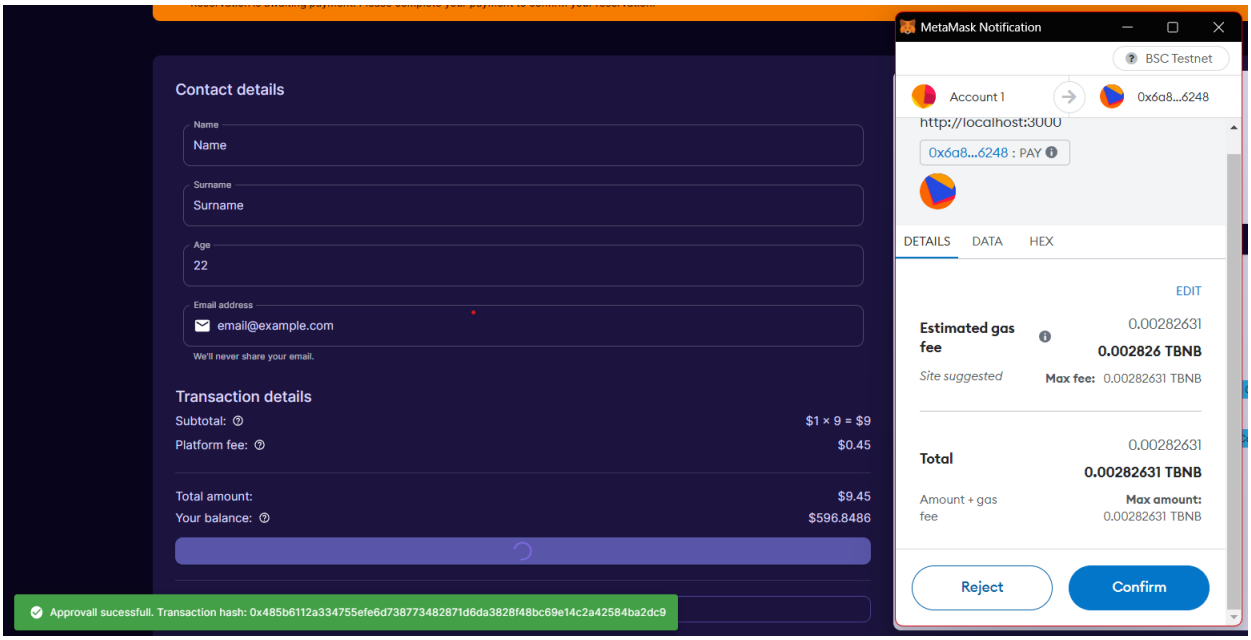


Figure 27. Smart contract interaction

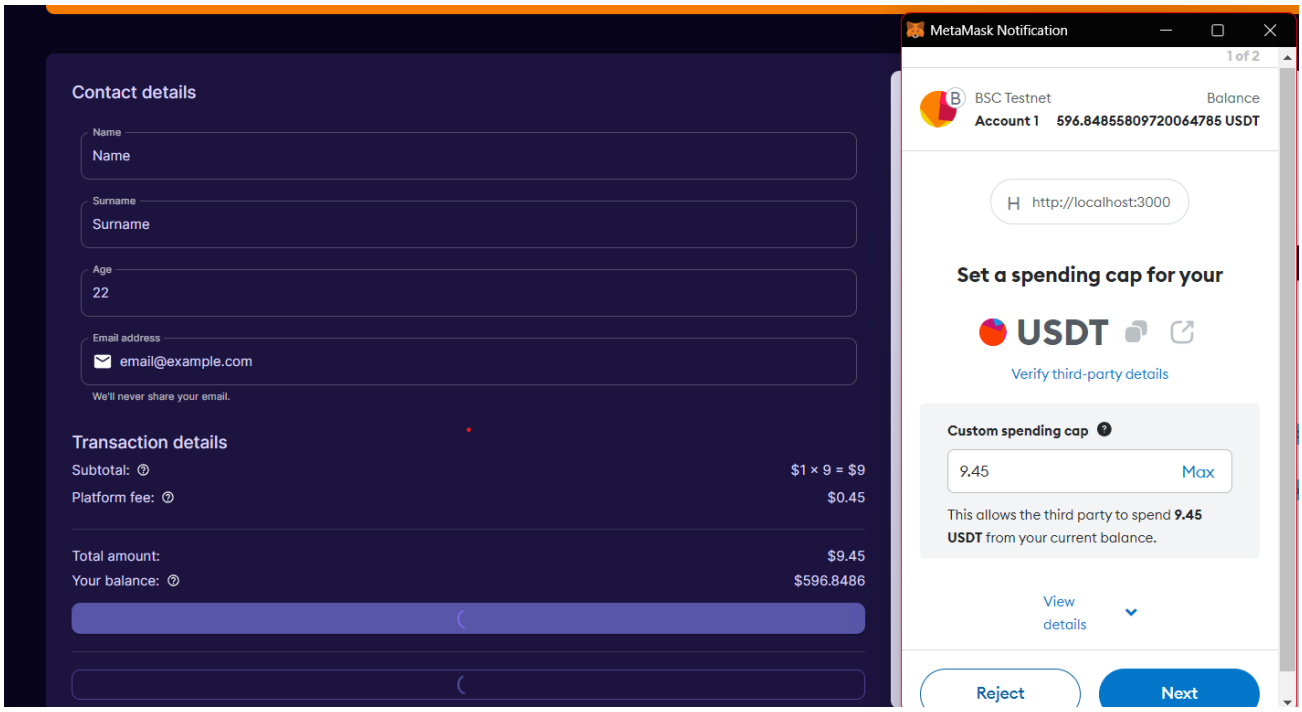


Figure 28. Allowance setting interaction

# References


. Arstechnica (2023). **Mysterious leak of Booking.com reservation data is being used to scam customers**. [Accessed May 1, 2023]. URL: https://arstechnica.com/information-technology/2023/02/mysterious-leak-of-booking-com-reservation-data-is-being-used-to-scam-customers/.

. BscScan (2022). **BNB Smart Chain Average Gas Price Chart**. [Accessed May 1, 2023]. URL: https://bscscan.com/chart/gasprice.

. CNBC (2021). **Nearly all of the $600 million stolen in a huge crypto heist has been returned — but there's a catch**. [Accessed May 1, 2023]. URL: https://www.cnbc.com/2021/08/13/poly-network-hack-nearly-all-of-600-million-in-crypto-returned.html.

. Computerweekly (2020). **Airbnb hosts' account data exposed in internal leak**. [Accessed May 1, 2023]. URL: https://www.computerweekly.com/news/252489702/Airbnb-hosts-account-data-exposed-in-internal-leak.

. NerdWallet (2021). **Credit Card Processing Fees: What You Need to Know**. [Accessed May 1, 2023]. URL: https://www.nerdwallet.com/article/small-business/credit-card-processing-fees.

. Paypal (2022). **PayPal Consumer Fees**. [Accessed May 1, 2023]. URL: https://www.paypal.com/us/webapps/mpp/paypal-fees.

. Web3Auth (2023). **How Web3Auth Works?** [Accessed April 28, 2023]. URL: https://web3auth.io/docs/how-web3auth-works.