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

Course work, Mid-term 2

Tripio - A web3 travel accommodation website

Done by:
Deividas Bendaravičius

Supervisor:
Linas Būtėnas

Vilnius
2023

Contents

1 Summary

The core objective of the system is to provide users with a comprehensive and easy platform to search, compare, and book accommodations that meet their preferences and budget, thereby enhancing their travel experience. In addition to just being a booking accommodation, payments will be done via Blockchain transactions, with the possibility to "stream" money. Hence, the project is a Web3 integration into the existing eco-system today.

2 Introduction

This section is reserved for the introduction of the document, and will be filled in later

3 Web3

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).

3.1 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.

3.2 Transaction benefits

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.

3.2.1 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.

3.2.2 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 3.2.1 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.

3.3 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.

3.3.1 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)

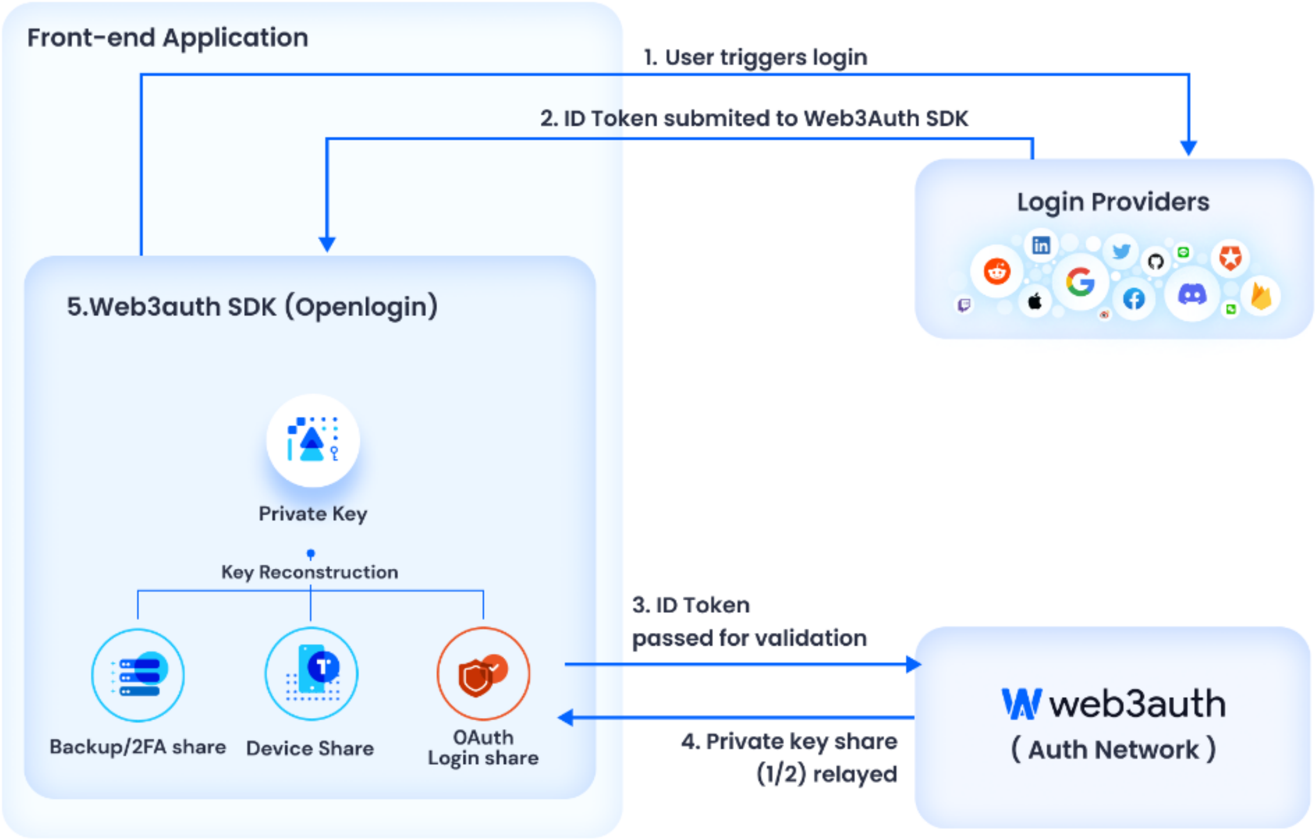


Figure 1. Web3Auth SDK flow

3.3.2 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).

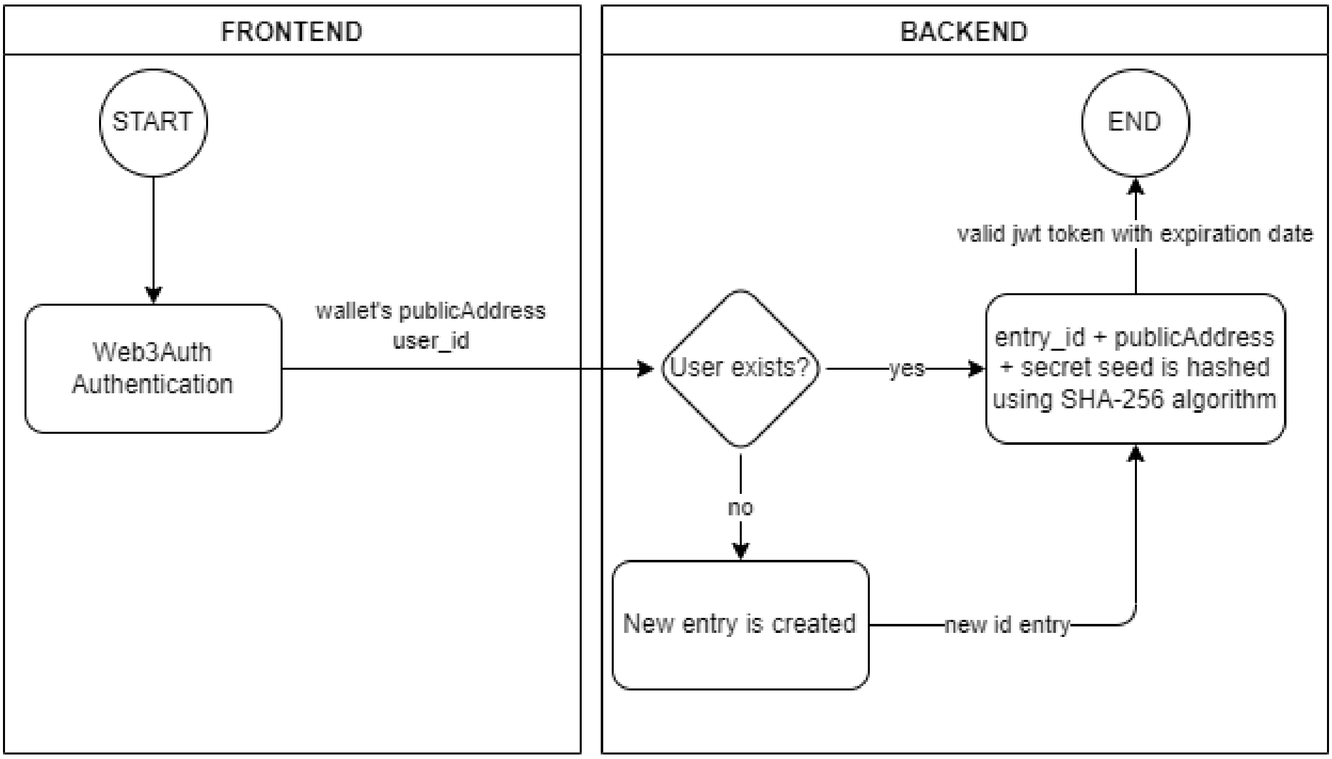


Figure 2. Authentication process

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.

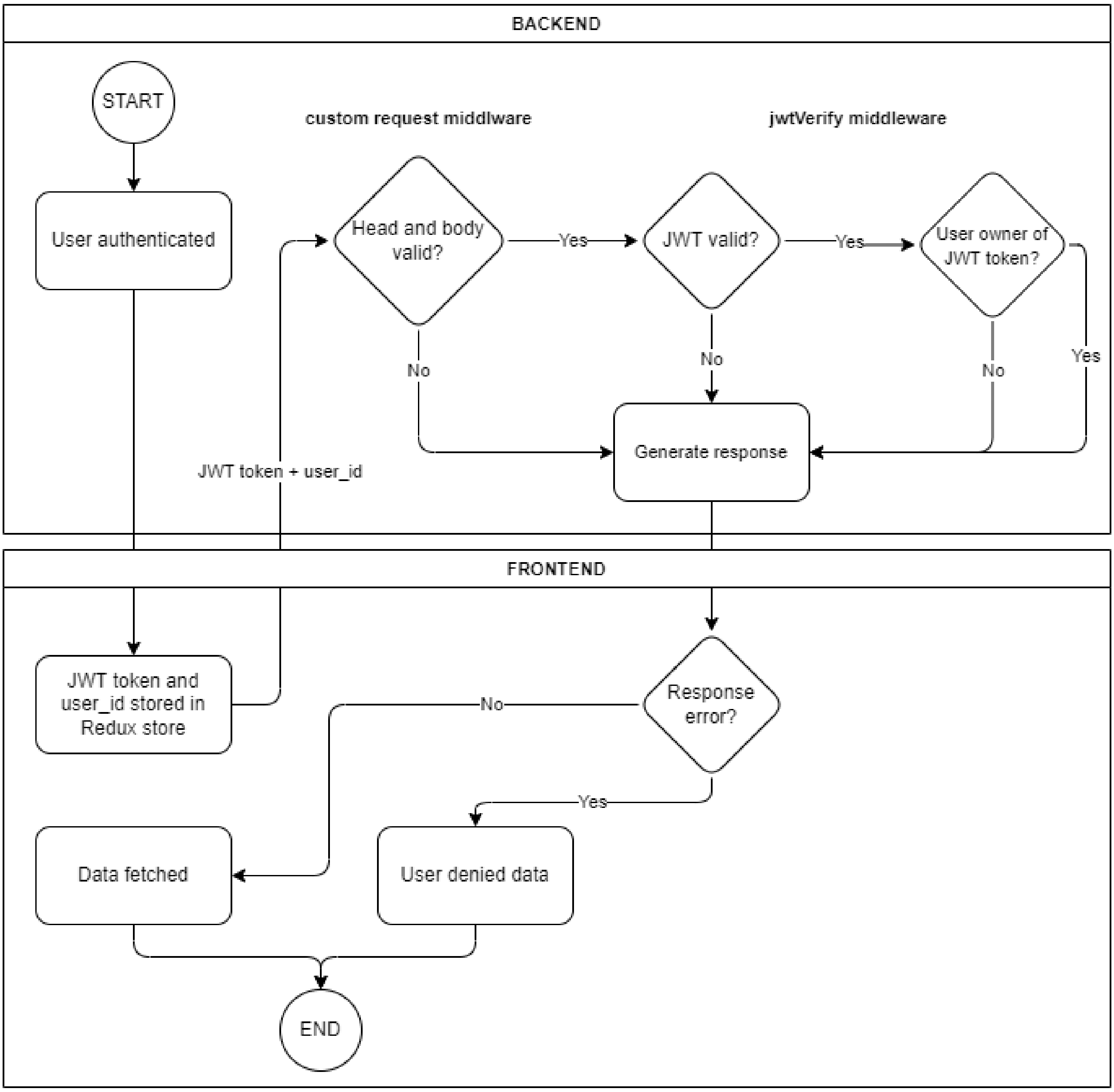


Figure 3. Protected route access Process

3.4 Payments

Work in progress....

4 Technologies

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

The MERN stack:

- Frontend: React.js + Typescript
- Backend: Node.js + Express.js + Typescript
- Database: MySQL
- 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.

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

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.

5 Non-functional requirements

SECURITY The app will be using HTTPS requests between the backend and frontend to ensure data encryption. The system will also handle sensitive user data such as login credentials, to combat the risk of a leak, all data will be hashed and nothing is going to be stored in plain text. The user accounts will be protected by requiring the user to login with chosen method (such as Social logins, Metamask, Torus wallet or Walletconnect). Accessing protected routes to the backend will require JWT authorization token, which is 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 be applying best practices and other HCI principals to make the user experience easy and pleasant to use. The components that will be used to build the UI are going to be easily recognizable, because of the popular MUI library that many websites already use.

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 that optimizes rendering based on the user's actions. This allows the application to be performant on any device.

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.

6 System architecture

6.1 Use cases

The diagram below showcases the use cases of the system. There are three actors at play. User is a simple user that is able to use the majority of the functionalities that the system brings. User (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 User (ADMIN) is a user that can delete properties, ban users and view reports of the regular users.

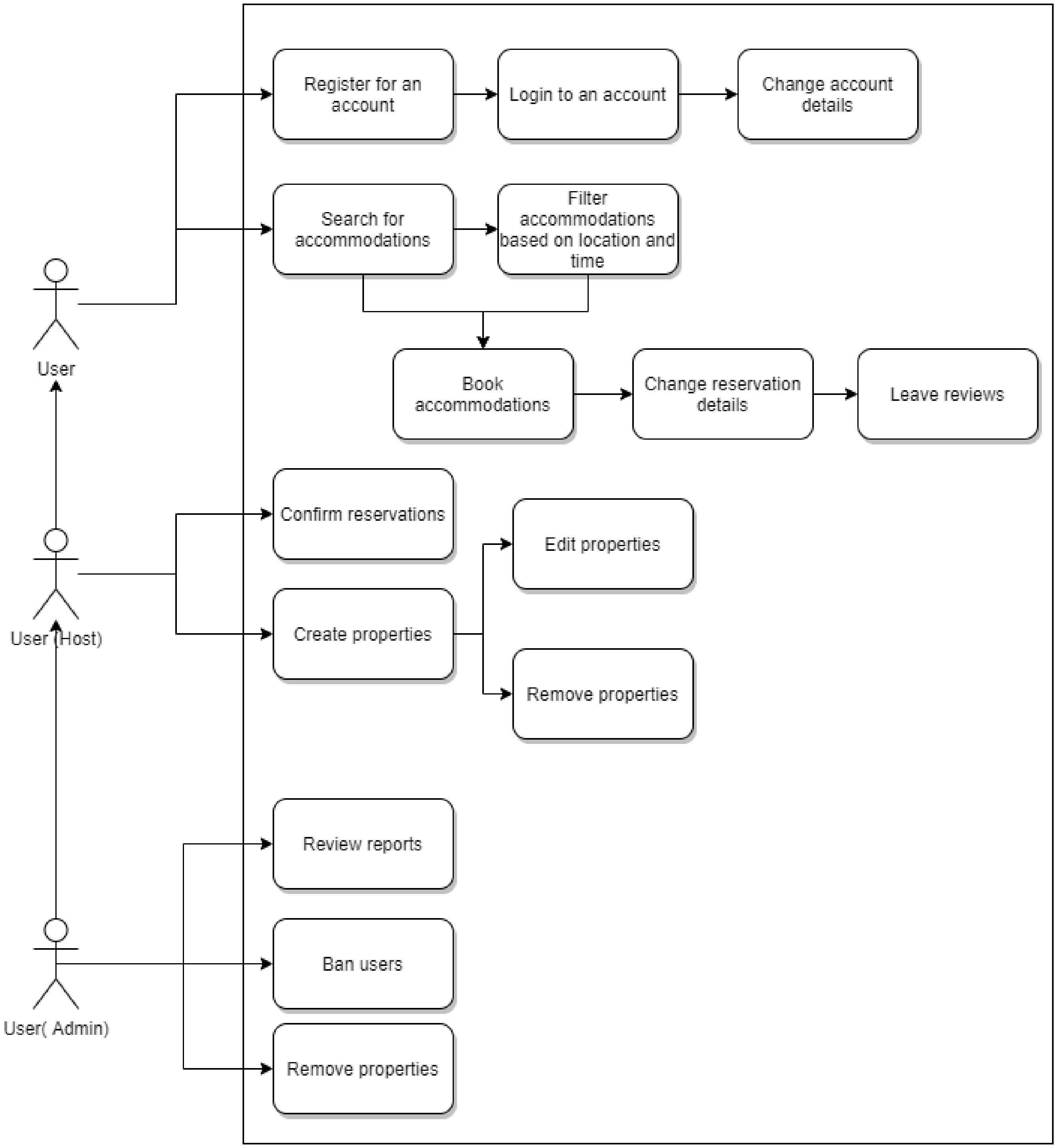


Figure 4. UML Use Case diagram

User:

- Register for an account - before using the website, a user has to create an account.
- Login to an account - in order to access listings, the user has to login on the website.
- Change account details - the user can change his account details, such as name, age, profile picture etc.
- Search for accommodations - the user is able to search for provided accommodations.
- Filter accommodations - the user is able to filter accommodations by location and availability, as well as price.
- Book accommodations - the user is able to book an accommodation.
- Change reservation details - the user is able to request a change of reservation details.
- Leave reviews - the user is able to leave reviews of the accommodation by the end of the stay.

User (Host). All of the above plus:

- Confirm reservations - the host user can confirm or decline reservations
- Create properties - the host user can create a property to list
- Edit properties - the host user can edit properties
- Remove properties - the user is able to delete owned properties from being listed.

6.2 Deployment

The deployment of the entire system is displayed in the diagram below. The system will consist of three parts:

- Web browser running the React.js website
- Cloud platform hosting two servers. Backend server to handle communication between the database and the frontend, running Node.js/Express.js. Database server storing data and listening to queries from the backend, running MySQL.

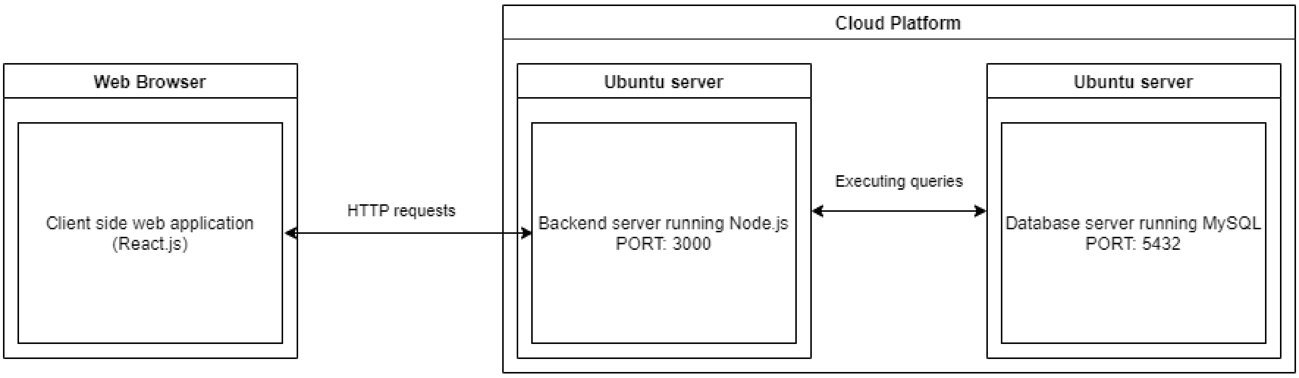
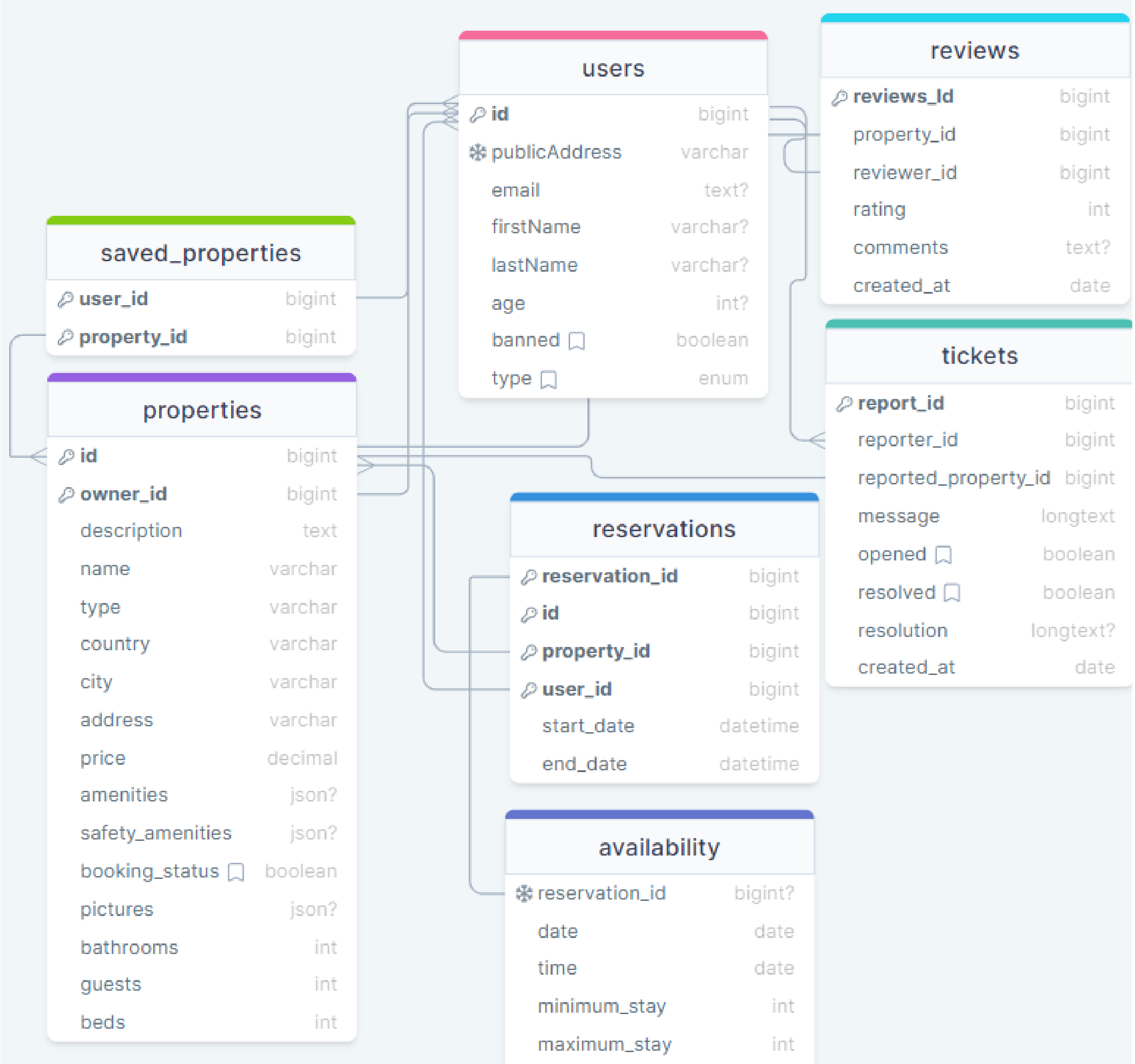

Cloud Platform
Web Browser
Client side web application (React.js)
HTTP requests
Ubuntu server
Backend server running Node.js PORT: 3000
Executing queries
Ubuntu server
Database server running MySQL PORT: 5432

Figure 5. UML Deployment diagram

6.3 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. Note, this is WIP(work-in-progress) design and might change as the system gets developed. Some tables will be split and some tables will be deprecated.



users	
id	bigint
publicAddress	varchar
email	text?
firstName	varchar?
lastName	varchar?
age	int?
banned	boolean
type	enum

reviews	
reviews_Id	bigint
property_id	bigint
reviewer_id	bigint
rating	int
comments	text?
created_at	date

saved_properties	
user_id	bigint
property_id	bigint

tickets	
report_id	bigint
reporter_id	bigint
reported_property_id	bigint
message	longtext
opened	boolean
resolved	boolean
resolution	longtext?
created_at	date

properties	
id	bigint
owner_id	bigint
description	text
name	varchar
type	varchar
country	varchar
city	varchar
address	varchar
price	decimal
amenities	json?
safety_amenities	json?
booking_status	boolean
pictures	json?
bathrooms	int
guests	int
beds	int

reservations	
reservation_id	bigint
id	bigint
property_id	bigint
user_id	bigint
start_date	datetime
end_date	datetime

availability	
reservation_id	bigint?
date	date
time	date
minimum_stay	int
maximum_stay	int

Figure 6. Database Relational model

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
  - password - hashed user password
  - 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
  - pictures - property pictures
  - bathrooms - property size information
  - guests - property size information
  - beds - property size information

- saved_properties
  - property_id - saved property id
  - user_id - user that saved the property id

- reservations
  - reservation_id - reservation id
  - property_id - property id
  - user_id - user that reserved the property id
  - start_date - start date of reservation
  - end_date - end date of reservation

- availability (MIGHT BE DEPRECATED)
  - reservation_id - reservation id

  - date - beginning date
  - time - duration of stay

- tickets

  - report_id - unique id of a report
  - reported_id - id of the user that reported
  - reported_property_id - property that was reported
  - message - report message
  - opened - boolean value if the ticket was viewed by an admin
  - resolved - boolean value if the ticket was resolved
  - resolution - admin message about the resolution

- reviews

  - reviews_id - unique id of a review
  - property_id - property id that was reviewed
  - reviewer_id - user id that reviewed the property
  - rating - rating (0-5)
  - comments - user left comments
  - created_at - date of the review

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.