HomeHire: A Blockchain-based Housing Rental System

Barak Levy, Onn Rengingad

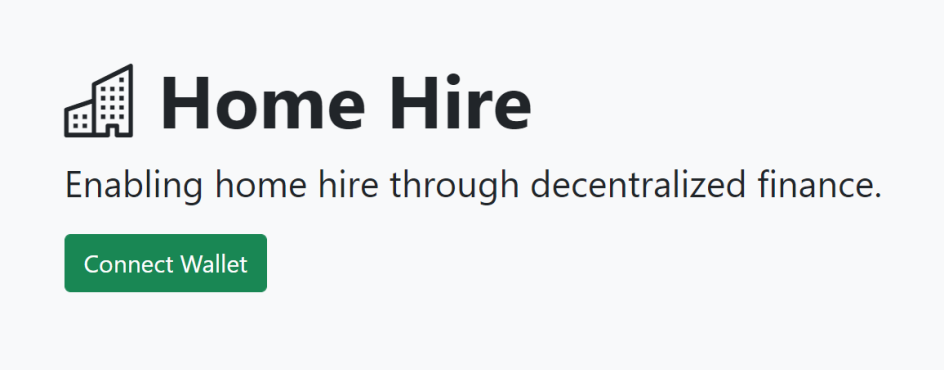

Figure 1: An image of our site

The full project code, with instructions of how to run the demo application, can be accessed here.

1 Introduction

With real estate prices soaring to record heights, it has never been more difficult to qualify for own apartment. The result to this situation is high increase in house renting. The old, naive and cumbersome mechanism of signing face-to-face contracts are seems outdated these days. Our project proposes a blockchain-based solution for renting system to achieve peer-to-peer sharing of listings information in an intermediary-free (gas cost) manner. The smart contract converts traditional lease agreements into smart lease contracts, improve the efficiency of the leasing process, and store transaction records for the leasing process to provide legal protection for tenants and landlords.

Our smart contract is simple, but powerful. To list property on the 'HomeHire' network, landlords must first be verified by the contract administrator. Once approved, they can add their rental property (representing a trusted administrator). Our contract support the following functions:

- add_landlord.
- add_home.
- change_renter.
- change_home_rent.
- delete_home.
- pay_rent.

Multiple require specifiers added to the Smart Contract in order to enforce correctness.

2 Tech Stack - Application Infrastructure

2.1 Solidity

The smart contract written in Solidity programming language. Solidity used for implementing smart contracts on various blockchain platforms, most notably, Ethereum. Compiler used: 0.6.0.

2.2 Python's FastAPI

We are using Python's FastAPI to create the API endpoints and the Web3.py library to preparethe Ethereum transactions.

2.3 PostgreSQL

We are using basic PostgreSQL database to store some off-chain data, like URL's to images of the homes.

2.4 Metamask wallet

Rather than asking the user to hard-code their private keys into the application itself (a SERIOUS security risk), we sign transactions using the popular Metamask browser extension.

3 Acquaintance with the contract

If you're familiar with object-oriented programming languages, most of the code should be pretty straightforward. I would like to highlight a couple of key functions:

- **struct Home {**
  address renter; Tenant address.
  address owner; Landlord address.
  uint256 homeRent; Price of the property in Wei.
  uint256 roomNum; Number of rooms of the property.
  uint256 monthsToPay; How many months left to pay.
  string dateOfStart; Starting date of the tenancy.
  string dateOfEnd; finishing date of the tenancy.
  string garage; Is parking included.
  string elevator; If there is elevator.
  }
  This struct will store on the blockchain.
- **mapping (address = bool) public Landlords;** Mapping used to track relationship between users and approved Landlords.
- **mapping (address = Home) public Homes;** Mapping used to track relationship between tenant to the exclusive property he rents.
- **function addLandlord(address _landlord) public.** Function used to add Landlord - Only the contract deployer can perfrom this action.
- **function addHome(uint256 _homeRent, uint256 _monthsToPay, address _renter, string memory _dateOfStart, string memory _dateOfEnd, string memory _elevator, string memory _garage, uint256 _roomNum) public.** Function used to add home for rental. Each property is tenant-exclussive and only approved landlord can add home.
- **function changeRenter(address _oldRenter, address _newRenter, string memory _dateOfStart, string memory _dateOfEnd, uint256 _monthsToPay) public.** Function used to update the property tenant. Each property is tenant-exclusive and only approved landlord can update home.
- **function changeHomeRent(address _renter, uint256 _newHomeRent) public.** Function used to update the property rent-price. Each property is tenant-exclusive and only approved landlord can update rent-price.
- **function deleteHome(address _renter) public.** Function used to delete property. Only the contract deployer can delete a property. Used to enforce unprotected tenancy.

- **function payRent(address payable _to) public payable.** Function used to pay a rent by the tenent to the appropriate landlord.
- **function getHome(address _renter) public view returns(address, address, uint256, uint256, uint256, string memory, string memory, string memory, string memory).** Getter function used to retrieve a property given tenant address(one-to-one relation).

4 Other Functionalities

I would like to highlight a function that one can interact throughout the application but doesn't part of the contract:

1. **WEI to ETH calculator**, this is a simple calculator that converts WEI to ETH. Soldity does not support floating point numbers so the smallest ETH unit of measure is used to pay in transactions.

Figure 2: An image of our calculator

4.1 Good luck!