Decentralized Book Rental Platform - Project Report

Team-name: Cryptohives

Vangapally PranavaReddy - mc230041037,

Annamareddi Suhitha - cse230001008,

Polathala Bhavana - mc230041026,

Kommireddy Jayanthi - cse230001041,

Akella Akhila - cse230001005,

Parimi Sunitha - cse230001061.

1. Introduction

This project implements a **Decentralized Book Rental Platform** using blockchain technology. The platform allows book owners to list their books for rent, and renters to rent and return books through a transparent, trustless system. All operations such as listing, renting, returning, and refunds are executed via smart contracts deployed on a local Ethereum blockchain (Ganache). The frontend is developed using React.js and interacts with the blockchain via Web3.js. MetaMask is used for transaction approvals.

2. Project Overview

Core Functionalities:

- Owner authentication to list books
- Renter authentication to rent books
- Smart contracts governing book availability, rent transactions, returns, and refund calculations
- Transparent payment handling through MetaMask
- Penalty calculations for late returns

Key Technologies Used:

- **Solidity**: Smart contract development
- **React.js**: Frontend development
- **Web3.js**: Blockchain interaction
- **Ganache**: Local Ethereum blockchain
- **MetaMask**: Wallet for transaction signing

3. Architecture

- **Smart Contract (Solidity)**:

    - Manages book listings, rentals, returns, unlisting, and refunds.
    - Enforces access control to restrict listing and unlisting to owners.
    - Handles rent payments and secure storage of deposits.
- **Frontend (React.js)**:
    - Interfaces for dashboard views of owners and renters.
    - Form inputs for listing and renting books.
    - Button-triggered MetaMask transactions for all blockchain interactions.
- **Blockchain (Ganache)**:
    - Provides a local blockchain environment for development and testing.
- **Wallet (MetaMask)**:
    - Connects users to blockchain network.
    - Prompts for transaction approvals.

4. Workflow

- **Owner Listing a Book:**
    - Enters book details and submits form.
    - Transaction initiated in MetaMask to add book to the blockchain.
- **Renter Renting a Book:**
    - Clicks the "Rent" button.
    - Confirms payment transaction (Daily Price + Deposit) through MetaMask.
- **Owner Unlisting a Book:**
    - Can unlist a book only if it is not currently rented.
    - Confirms the unlist transaction via MetaMask.
- **Renter Returning a Book:**
    - Clicks the "Return" button.
    - Penalty for late return is calculated automatically.
    - Refund amount is computed and shown.
- **Renter Withdrawing Refund:**
    - Clicks the "Withdraw" button.
    - Confirms refund transaction to receive the remaining deposit.

5. Edge Case Handling

- **Renting an Unavailable Book:** Solidity checks book availability and reverts; App.js catches the error and alerts "Book not available".
- **Double Rental Attempts:** Solidity prevents double renting by ensuring proper conditions; App.js disables rent button and alerts on error.
- **Reentrancy Attack:** OpenZeppelin's ReentrancyGuard prevents reentrancy on rentItem, returnItem, and withdrawRefund.
- **Early Returns:** Solidity allows early returns with partial minute rounding and accurate refund adjustment.
- **Insufficient Payment:** Solidity checks payment sufficiency; App.js shows alert if payment is insufficient.

- **Failed Transactions:** All critical actions are wrapped in try-catch blocks in App.js with appropriate alerts for failures.

6. Optimizations Implemented

- **Efficient Storage**: Only essential book information is stored on-chain, minimizing gas usage.
- **Minimal State Changes**: State updates occur only when necessary to reduce transaction costs.
- **Batch Updates Avoided**: Separate transactions for rent, return, unlist operations prevent unnecessary bundling.
- **Frontend Caching**: Book list is refreshed only after significant operations to reduce unnecessary blockchain reads.

7. Security Measures

- **Access Control:**
  - Only registered owners can list and unlist books.
  - Renters are restricted to renting, returning, and withdrawing operations.
- **Deposit System:**
  - Renters pay a deposit to ensure the return of the book.
  - Penalties are deducted automatically based on the return delay.
- **Reentrancy Protection:**
  - Smart contract design separates state change and fund transfer logic.
  - Optionally, OpenZeppelin's ReentrancyGuard can be used for production deployment.
- **Safe Fund Handling:**
  - Refunds and deposits are handled separately through withdrawal pattern, preventing direct transfer vulnerabilities.
- **MetaMask Transaction Confirmation:**
  - Every action requiring blockchain changes prompts explicit confirmation in MetaMask.

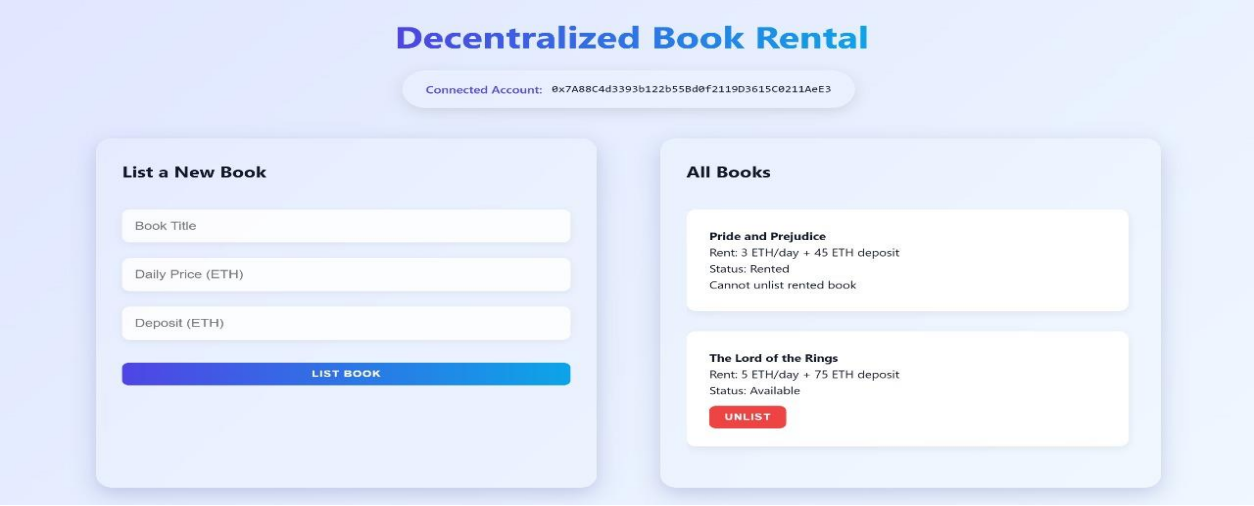
8. Testing

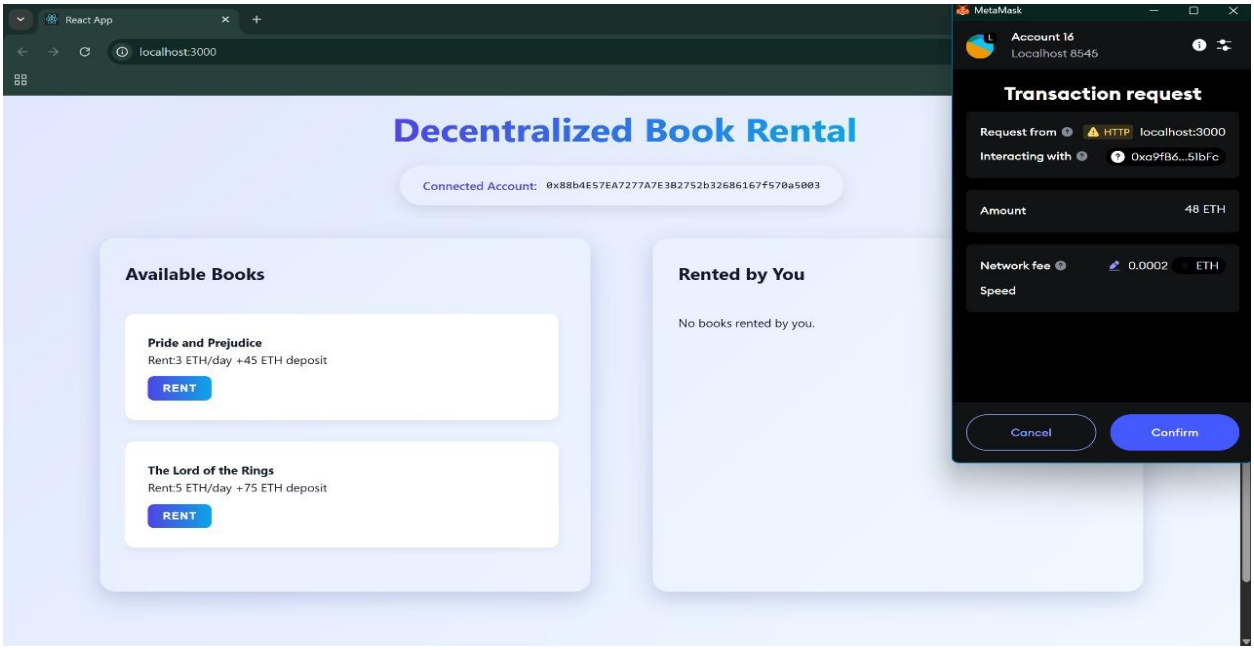
Test suite checks the core functionalities of the BookRental contract, including book listing, renting, returning, penalty application, refund withdrawal, unlisting, and handling various error scenarios like insufficient payment or double rentals.

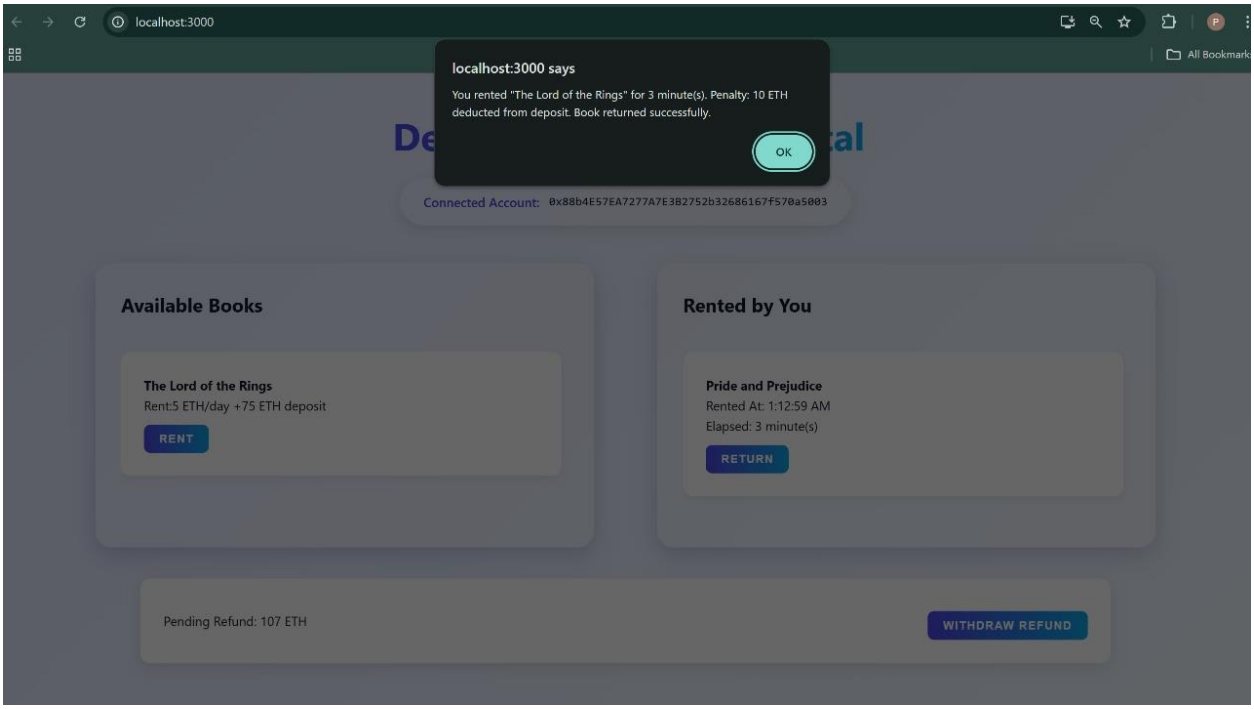
```
Contract: BookRental
  ✔allow book listing (122ms)
  ✔rent and return a book within 1 minute with full refund (451ms)
  ✔apply penaltyfee if returned after 1 minutes (511ms)
  ✔result in no refund after 5+ minutes (521ms)
  ✔allow withdrawal of refund (716ms)
  ✔allow owner to unlist an available book (206ms)
  ✔reject unlisting if book is rented (441ms)
  ✔reject double rental attempts (421ms)
  ✔reject insufficient payments (117ms)


9 passing (4s)
```

9. Interpretations

This decentralized application demonstrates how blockchain technology can manage real-world peer-to-peer rentals securely without a centralized authority. It shows user-role management, transaction security, penalty enforcement, and transparent refund processing in a simple and user-friendly environment.







10. Future Enhancements

- Adding real-time event listeners to update UI immediately after transactions.
- Implementing dynamic penalty policies.
- Deploying on public testnets (e.g., Goerli) and mainnets.
- Multi-owner management and full admin dashboards.

11. Conclusion

The Book Rental DApp successfully showcases a secure, transparent, and user-driven rental system using blockchain. It emphasizes the advantages of decentralized platforms in replacing traditional intermediaries and ensuring fair financial operations.