The UNIVERSITY of OKLAHOMA
Gallogly College of Engineering
School of Computer Science

Decentralized GPU Rental Platform

Narayana Phani Charan Nimmagadda (113642485)
Sai Madhukar Vanam (113649570)

Contents

1 Introduction

The Decentralized GPU Rental Platform is a blockchain-based marketplace designed to facilitate trustless GPU rentals with automated performance verification and fair payment distribution. By leveraging Ethereum smart contracts, Chainlink Functions, and IPFS (via Pinata), the platform addresses key challenges in traditional GPU rental systems, such as lack of transparency, trust issues, and manual verification processes. Users can rent GPUs through a web interface, with payments held in escrow until performance is verified, ensuring providers are compensated based on actual delivered performance while users are refunded for any underperformance.

The complete source code for the platform is available on GitHub: `https://github.com/NNPhaniCharan/RentGPU`

A demonstration video explaining the platform's features and functionality is available at: `https://youtu.be/YourVideoLinkHere`

This report outlines the project's background, methodology, technical implementation, results, and conclusions, providing a comprehensive overview of the platform's design and functionality.

2 Background

2.1 Problem Statement

Traditional GPU rental systems often rely on centralized platforms, which can lead to:

- Lack of transparency in GPU performance claims.
- Disputes over payment due to unverifiable performance.
- High fees and manual verification processes.

2.2 Solution

The Decentralized GPU Rental Platform introduces a trustless, automated system where:

- Payments are held in a smart contract escrow.
- GPU performance is verified using Chainlink Functions, comparing actual performance (uploaded to IPFS) against promised specifications.
- Payments are distributed proportionally based on a fulfillment percentage, ensuring fairness.

2.3 Relevance

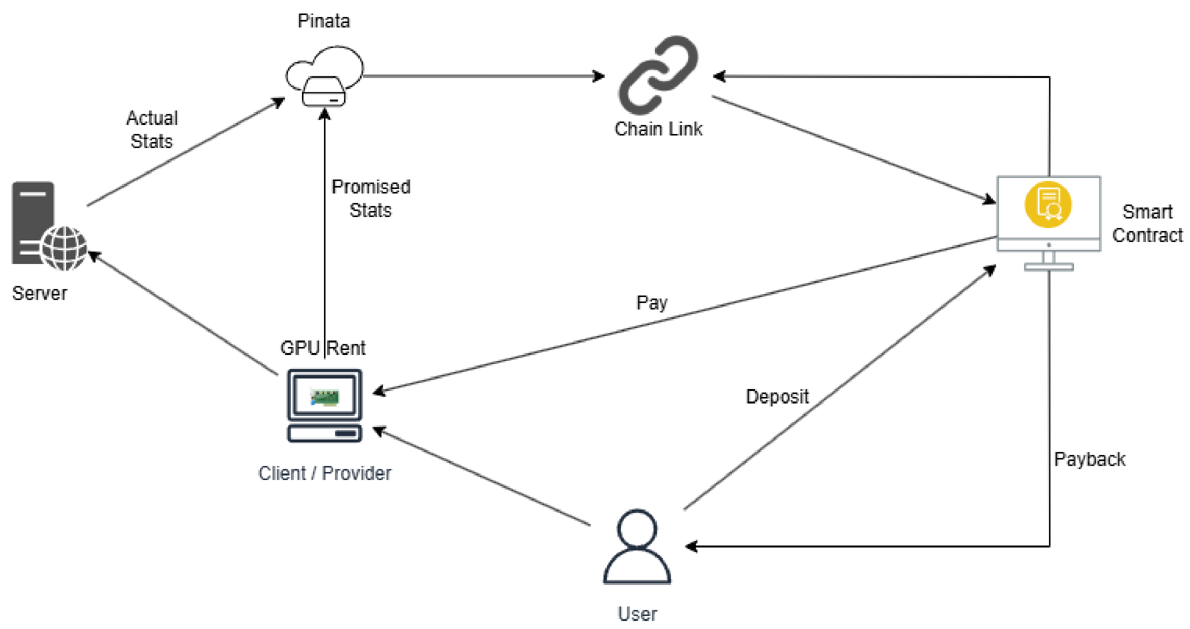
This platform is particularly relevant for industries requiring high-performance computing, such as machine learning, gaming, and cryptocurrency mining, where GPU performance directly impacts outcomes. By decentralizing the rental process, the platform reduces reliance on intermediaries and enhances trust through blockchain technology.

3 Methodology

3.1 System Architecture

The platform's architecture integrates multiple components to ensure a seamless rental and verification process. The key components are:

- **Frontend:** A React.js-based web interface for user interaction, built with Bootstrap for responsiveness.
- **Smart Contract:** Deployed on the Ethereum Sepolia Testnet, manages escrow, verification, and payment distribution.
- **Chainlink Functions:** Provides oracle services to fetch and compare GPU performance statistics.
- **IPFS (Pinata):** Stores immutable GPU specifications and performance metrics.
- **Client/Provider:** Supplies GPU resources for rental.
- **User:** Rents GPUs and provides ETH deposits.



System architecture illustrating the interaction between components.

3.2 Technology Stack

- **Blockchain:** Ethereum (Sepolia Testnet) for decentralized transaction management.
- **Smart Contracts:** Written in Solidity, deployed at address `0xE590ff0E4FB5fCE671053ED5091F215204F49433`
- **Frontend:** React.js, Web3.js/Ethers.js for blockchain interaction.

- **External Services:**
  - Chainlink Functions (Subscription ID: `4553`) for performance verification.
  - IPFS via Pinata for decentralized storage.

3.3 Workflow

The platform operates in three phases:

1. **Rental Process:**
   - User selects a GPU and pays in ETH.
   - Payment held in smart contract escrow.
   - Provider uploads promised performance metrics to IPFS.
2. **Verification Process:**
   - Provider runs GPU, uploads actual stats to IPFS.
   - Chainlink compares promised vs. actual stats.
   - Smart contract records fulfillment percentage.
3. **Payment Distribution:**
   - Payment disbursed based on fulfillment percentage.
   - Refunds issued for underperformance.

3.4 Setup & Installation

1. Clone the repository: `git clone https://github.com/NNPhaniCharan/RentGPU.git`
2. Install frontend dependencies: `cd frontend && npm install`
3. Configure environment variables in `frontend/.env`:
```
REACT_APP_PINATA_JWT=your_pinata_jwt
REACT_APP_GATEWAY_URL=your_pinata_gateway_url
REACT_APP_CONTRACT_ADDRESS=0xE590ff0E4FB5fCE671053ED5091F215204F49433
```
4. Start the development server: `npm start`
5. Connect MetaMask wallet with Sepolia ETH.

4 Results

4.1 Implementation Outcomes

The platform was successfully deployed on Sepolia Testnet. Key outcomes:

- **Automated Verification:** Chainlink Functions enabled trustless verification.
- **Fair Payments:** Payments/refunds based on fulfillment percentage.
- **Decentralized Storage:** IPFS ensured transparent data storage.

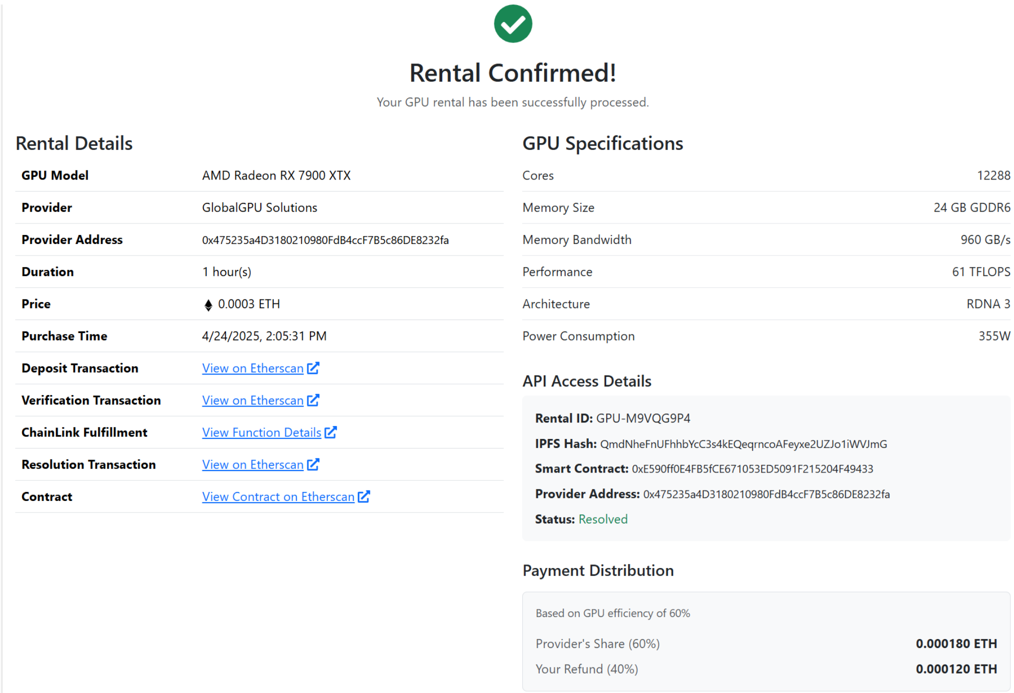
4.2 Performance Metrics

The following table summarizes GPU rental test cases, showing how the platform calculates fulfillment percentages and distributes payments based on actual GPU efficiency.
The following table summarizes GPU rental test cases, showing how the platform calculates fulfillment percentages and distributes payments based on actual GPU efficiency.

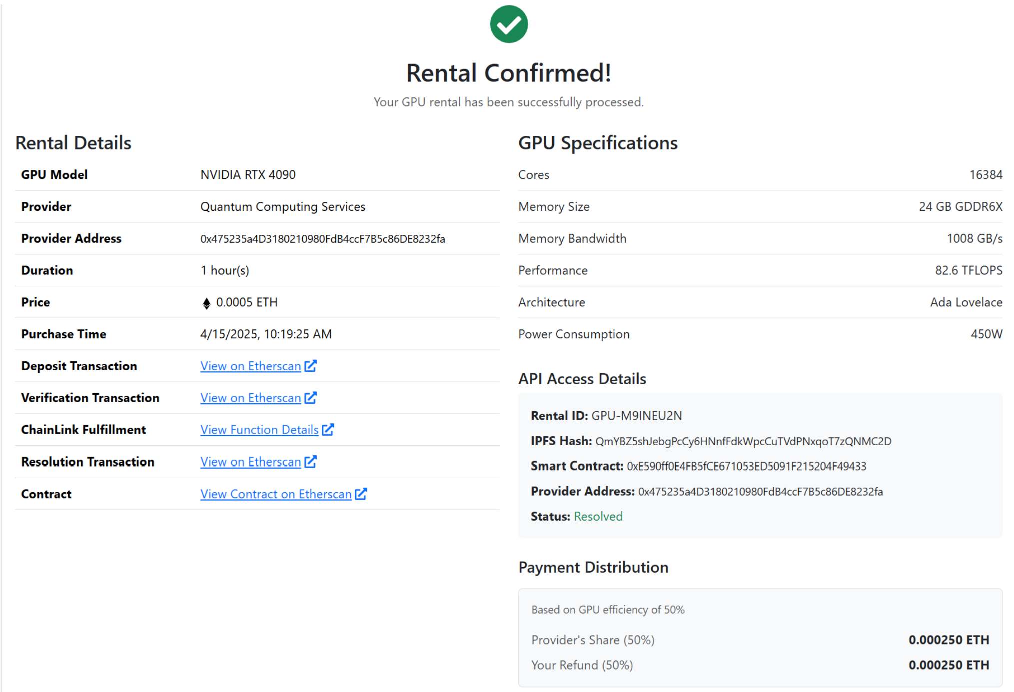
Test Case	GPU Model	Fulfillment %	Payment Distributed
1	AMD Radeon RX 7900	60%	60% to Provider
2	NVIDIA RTX 4090	50%	50% to Provider
3	NVIDIA RTX 4080	70%	70% to Provider

Table 1: GPU Rental Test Cases and Payment Distribution

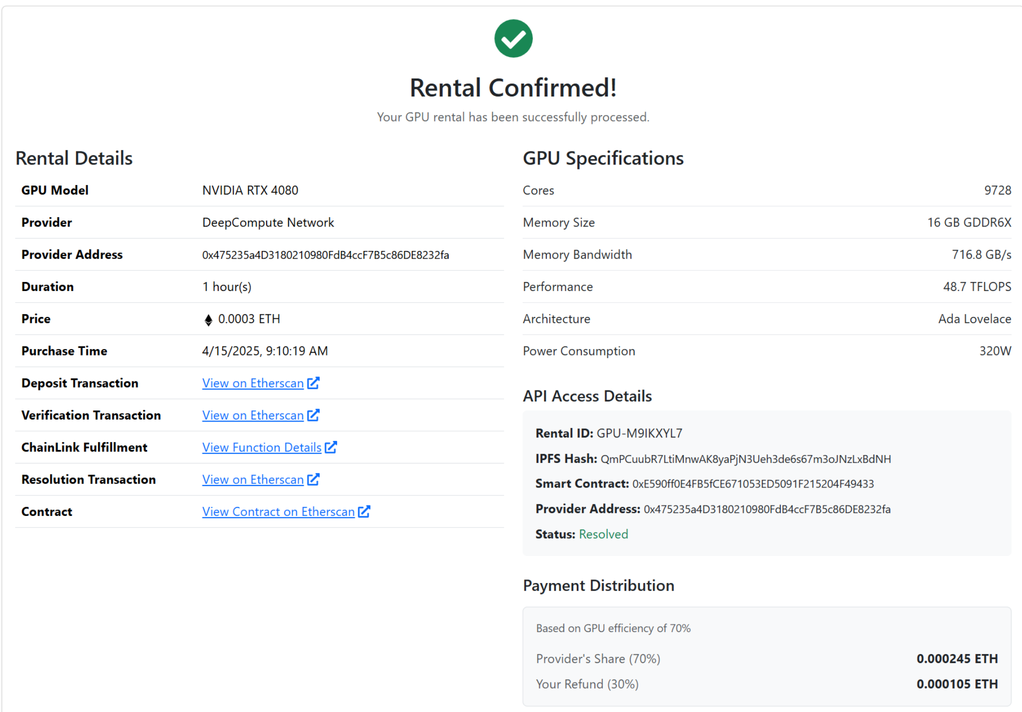
Below are the transaction confirmations for each test case:



GPU Rental Confirmation 1: AMD Radeon RX 7900 XTX (60% Efficiency)



GPU Rental Confirmation 2: NVIDIA RTX 4090 (50% Efficiency)



GPU Rental Confirmation 3: NVIDIA RTX 4080 (70% Efficiency)

5 Team Contributions

The project was collaboratively developed with distinct roles and responsibilities:

- **Narayana Phani Charan Nimmagadda:** Frontend Development, Smart Contract Development, Chainlink Functions Integration, Backend Logic, Project Coordination.
- **Sai Madhukar Vanam:** Frontend Development, IPFS Integration, UI/UX Enhancements, Testing & Documentation.

Both members actively participated in system design discussions, troubleshooting, and final deployment activities.

6 Conclusion

The platform demonstrates a trustless, automated GPU rental solution by integrating Ethereum, Chainlink, and IPFS. It ensures fair, transparent transactions and is scalable for high-performance computing applications.

Future enhancements:

- Integration with layer-2 solutions to reduce gas fees and improve scalability.
- Real-time performance dashboards for better user experience and transparency.
- Automated provider onboarding with verification of GPU authenticity to prevent fraudulent listings.
- Cross-chain compatibility to enable GPU rentals on other blockchain networks beyond Ethereum, enhancing flexibility and reducing dependency on a single network.

7 References

1. Ethereum Foundation. (2023). *Ethereum Documentation*. Retrieved from `https://ethereum.org/en/developers/docs/`
2. Chainlink. (2023). *Chainlink Functions Documentation*. Retrieved from `https://docs.chain.link/chainlink-functions/`
3. Pinata. (2023). *IPFS Storage API Guide*. Retrieved from `https://docs.pinata.cloud/`
4. React.js. (2023). *React Documentation*. Retrieved from `https://react.dev/`