



*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

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Background</b>	<b>1</b>
2.1	Problem Statement . . . . .	1
2.2	Solution . . . . .	1
2.3	Relevance . . . . .	2
<b>3</b>	<b>Methodology</b>	<b>2</b>
3.1	System Architecture . . . . .	2
3.2	Technology Stack . . . . .	3
3.3	Workflow . . . . .	3
3.4	Setup & Installation . . . . .	3
<b>4</b>	<b>Step-by-Step Execution</b>	<b>4</b>
<b>5</b>	<b>Results</b>	<b>7</b>
5.1	Implementation Outcomes . . . . .	7
5.2	Performance Metrics . . . . .	7
<b>6</b>	<b>Team Contributions</b>	<b>9</b>
<b>7</b>	<b>Conclusion</b>	<b>9</b>
<b>8</b>	<b>References</b>	<b>9</b>

# 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 under-performance.

The live platform is deployed and accessible at: <https://rent-gpu-phi.vercel.app/>

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/W3chPSaXrwQ>

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

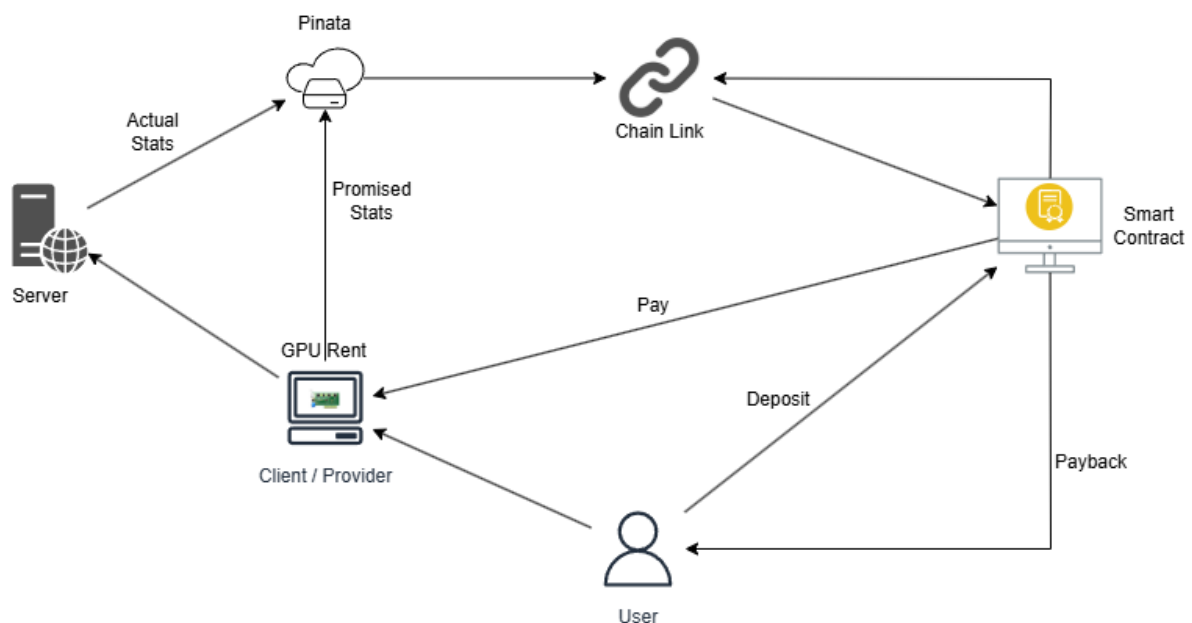
This platform is particularly relevant for industries requiring high-performance computing, such as machine learning, gaming, and cryptocurrency mining, where GPU performance directly impacts outcomes. By decentralizing the rental process, the platform reduces reliance on intermediaries and enhances 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 from all of the listed GPU's.
- Then selectes how many Hours the user wants the 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_GATEWAY_URL=your_pinata_gateway_url
REACT_APP_PINATA_API_KEY=your_pinata_api_key
REACT_APP_PINATA_SECRET_KEY=your_pinata_api_secret_key
REACT_APP_CONTRACT_ADDRESS=0xE590ff0E4FB5fCE671053ED5091F215204F49433
```

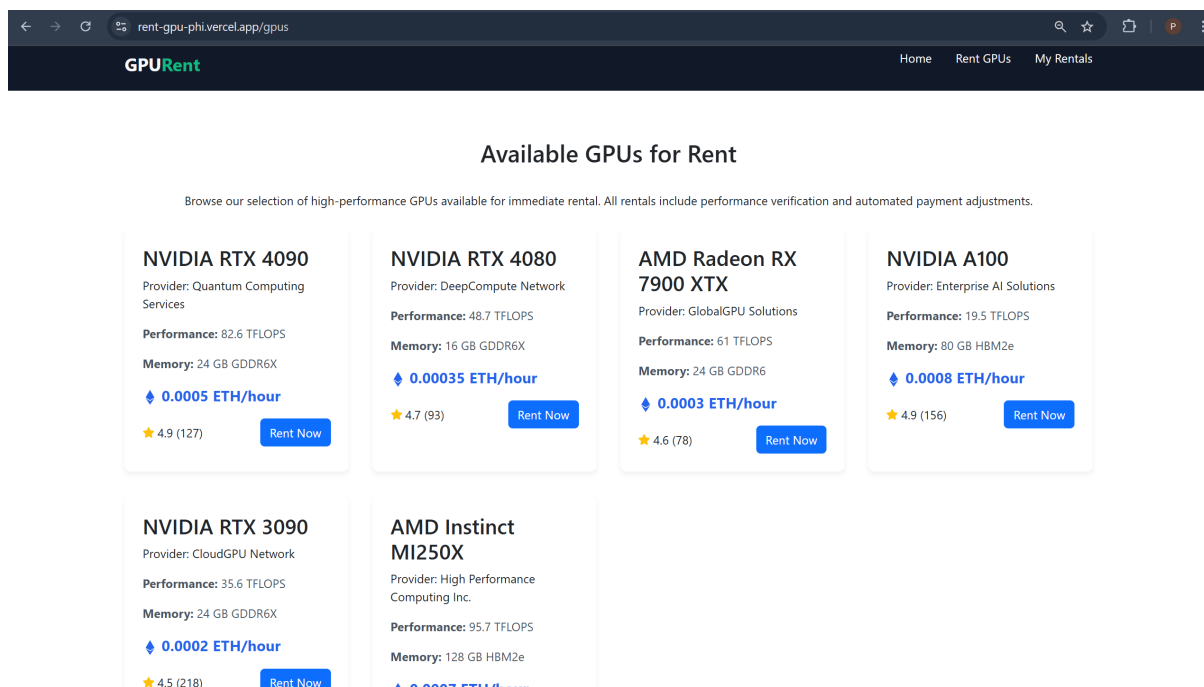
4. Start the development server: `npm start`
5. Connect MetaMask wallet with Sepolia ETH.

## 4 Step-by-Step Execution

This section describes the detailed step-by-step process of how a GPU rental is executed on the platform, from initiation to payment settlement.

### 1. User Browses Available GPUs:

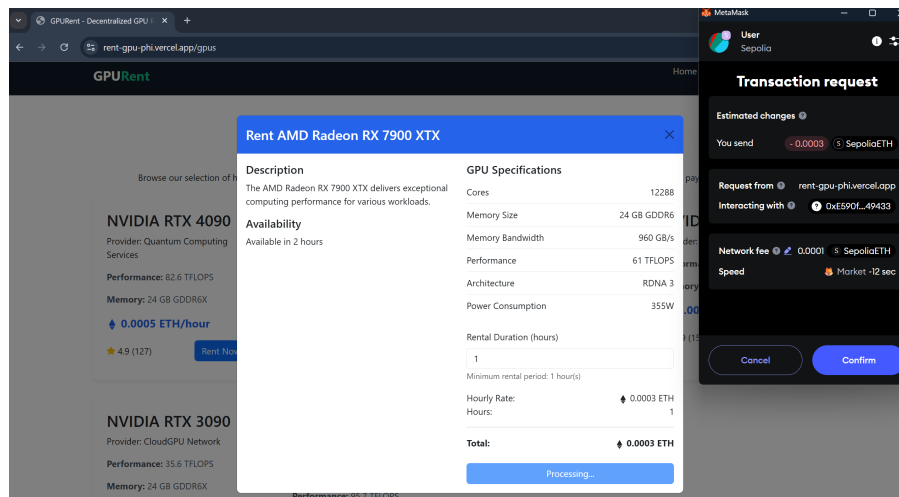
- User visits the web application and connects their MetaMask wallet to the Sepolia Testnet.
- The platform displays a list of available GPUs with specifications and hourly rental rates.



Step 1: User views available GPUs and pricing details

## 2. Initiate Rental:

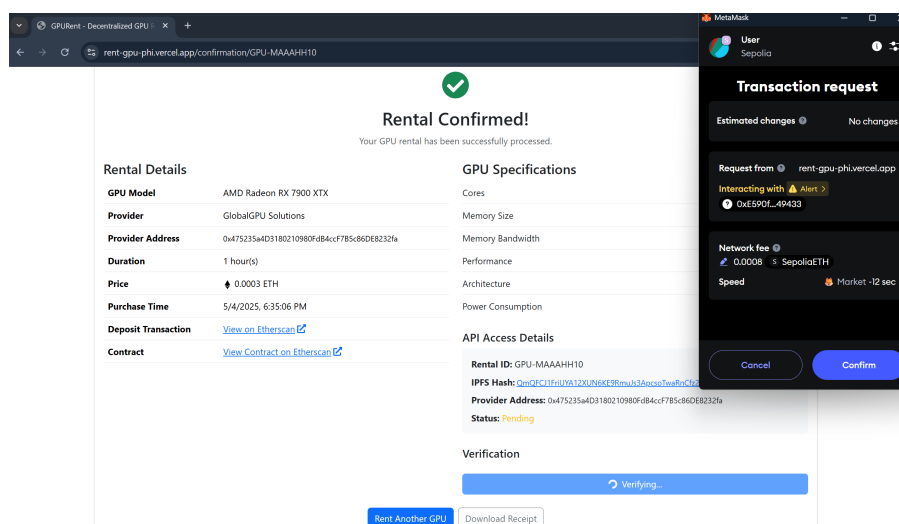
- User selects the desired GPU and rental duration.
- The required ETH deposit is calculated and a MetaMask transaction is prompted.
- The ETH is held in the smart contract escrow.



Step 2: User initiates rental and confirms payment via MetaMask

## 3. Verification Process:

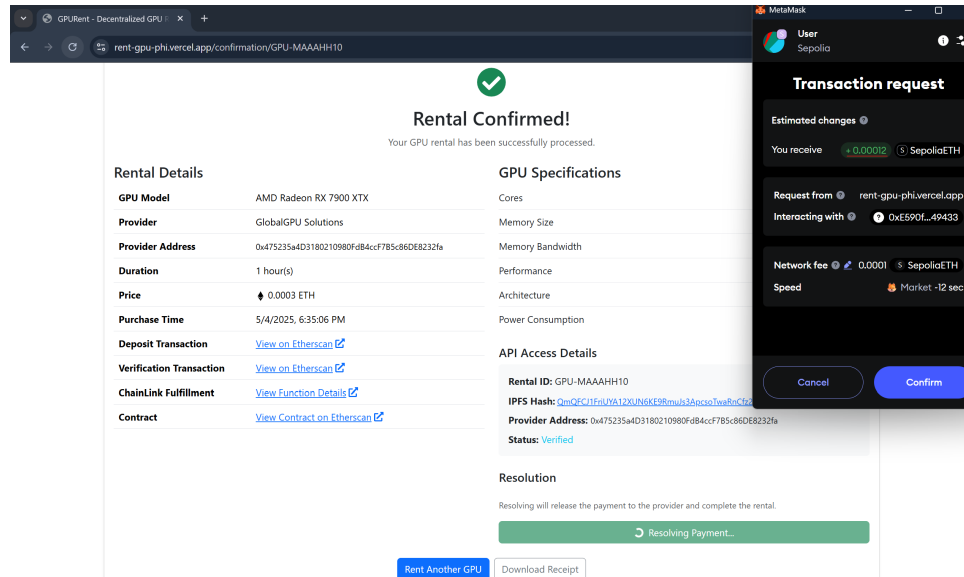
- Once the provider executes the task, actual performance is uploaded to IPFS.
- Chainlink Functions compare promised vs. actual metrics.
- Verification result and fulfillment percentage are recorded on-chain.



Step 3: Chainlink verification with fulfillment and contract update

#### 4. Payment Distribution:

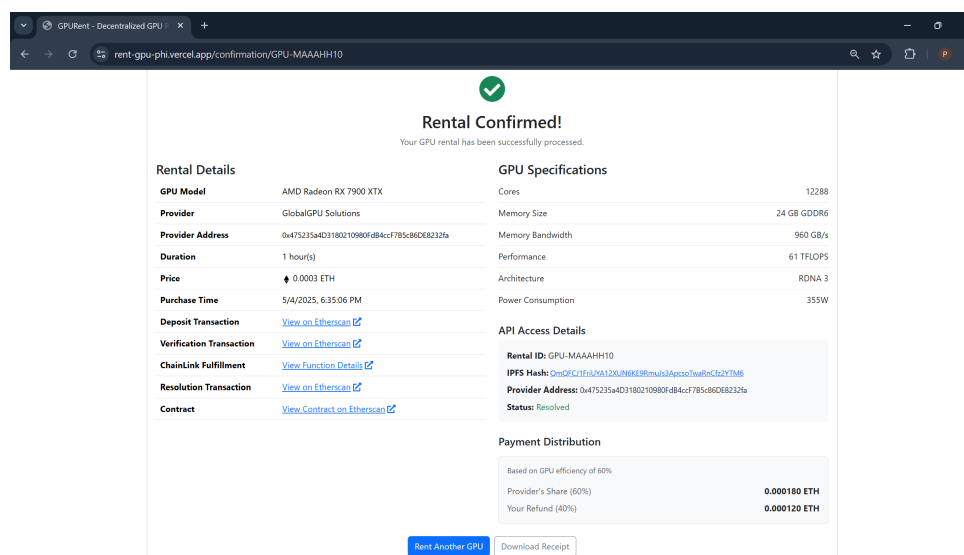
- Smart contract calculates and releases provider payment.
- Refunds are automatically sent to the user based on fulfillment percentage.



#### Step 4: Final transaction showing refund and provider's share

#### 5. Rental Summary:

- User can view all rental and transaction details.
- Includes on-chain verification links, performance results, and final receipts.



#### Step 5: Complete rental summary and payment breakdown



5 Results

5.1 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.

5.2 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:

Rental Confirmed!  
Your GPU rental has been successfully processed.

Rental Details

GPU Model

AMD Radeon RX 7900 XTX

Provider

GlobalGPU Solutions

Provider Address

0x475235a4D3180210980Fd84ccF7B5c86DE8232fa

Duration

1 hour(s)

Price

0.0003 ETH

Purchase Time

4/24/2025, 2:05:31 PM

Deposit Transaction

[View on Etherscan](#)

Verification Transaction

[View on Etherscan](#)

ChainLink Fulfillment

[View Function Details](#)

Resolution Transaction

[View on Etherscan](#)

Contract

[View Contract on Etherscan](#)

GPU Specifications

Cores

12288

Memory Size

24 GB GDDR6

Memory Bandwidth

960 GB/s

Performance

61 TFLOPS

Architecture

RDNA 3

Power Consumption

355W

API Access Details

Rental ID: GPU-M9VQG9P4

IPFS Hash: QmdRheFnUfHibYcC3s4KEQeqmcoAFeyxe2UZJo1WVvmG

Smart Contract: 0xE590F0E4FB5ICE671053ED5091F215204F49433

Provider Address: 0x475235a4D3180210980Fd84ccF7B5c86DE8232fa

Status: Resolved

Payment Distribution

Based on GPU efficiency of 60%


Provider's Share (60%)

0.000180 ETH

Your Refund (40%)

0.000120 ETH

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



### Rental Confirmed!

Your GPU rental has been successfully processed.

#### Rental Details

GPU Model	NVIDIA RTX 4090
Provider	Quantum Computing Services
Provider Address	0x475235a4D3180210980Fd84ccF7B5c86DE8232fa
Duration	1 hour(s)
Price	0.0005 ETH
Purchase Time	4/15/2025, 10:19:25 AM
Deposit Transaction	<a href="#">View on Etherscan</a>
Verification Transaction	<a href="#">View on Etherscan</a>
ChainLink Fulfillment	<a href="#">View Function Details</a>
Resolution Transaction	<a href="#">View on Etherscan</a>
Contract	<a href="#">View Contract on Etherscan</a>

#### GPU Specifications

Cores	16384
Memory Size	24 GB GDDR6X
Memory Bandwidth	1008 GB/s
Performance	82.6 TFLOPS
Architecture	Ada Lovelace
Power Consumption	450W

#### API Access Details

**Rental ID:** GPU-M9INEU2N

**IPFS Hash:** QmYBZ5shJebgPcCy6HnNfdkWpcCuTVdPNxqoT7zQNMCM2D

**Smart Contract:** 0xE590f0E4F85fCE671053ED5091F215204F49433

**Provider Address:** 0x475235a4D3180210980Fd84ccF7B5c86DE8232fa


**Status:** Resolved

#### Payment Distribution

Based on GPU efficiency of 50%

Provider's Share (50%)	0.000250 ETH
Your Refund (50%)	0.000250 ETH

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



### Rental Confirmed!

Your GPU rental has been successfully processed.

#### Rental Details

GPU Model	NVIDIA RTX 4080
Provider	DeepCompute Network
Provider Address	0x475235a4D3180210980Fd84ccF7B5c86DE8232fa
Duration	1 hour(s)
Price	0.0003 ETH
Purchase Time	4/15/2025, 9:10:19 AM
Deposit Transaction	<a href="#">View on Etherscan</a>
Verification Transaction	<a href="#">View on Etherscan</a>
ChainLink Fulfillment	<a href="#">View Function Details</a>
Resolution Transaction	<a href="#">View on Etherscan</a>
Contract	<a href="#">View Contract on Etherscan</a>

#### GPU Specifications

Cores	9728
Memory Size	16 GB GDDR6X
Memory Bandwidth	716.8 GB/s
Performance	48.7 TFLOPS
Architecture	Ada Lovelace
Power Consumption	320W

#### API Access Details

**Rental ID:** GPU-M9IKXYL7

**IPFS Hash:** QmPCuubR7LtiMmwAK8yaPjN3Ueh3de6s67m3oINzLxBdNH

**Smart Contract:** 0xE590f0E4F85fCE671053ED5091F215204F49433

**Provider Address:** 0x475235a4D3180210980Fd84ccF7B5c86DE8232fa

**Status:** Resolved

#### Payment Distribution

Based on GPU efficiency of 70%

Provider's Share (70%)	0.000245 ETH
Your Refund (30%)	0.000105 ETH

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

## 6 Team Contributions

The project was collaboratively developed with distinct roles and responsibilities:

- **Narayana Phani Charan Nimmagadda:** Frontend Development, Smart Contract Development, Chainlink Functions Integration, Project Coordination & Documentation.
- **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.

## 7 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.
- Cross-chain compatibility to enable GPU rentals on other blockchain networks beyond Ethereum, enhancing flexibility and reducing dependency on a single network.
- Prototype data only: For this project we did not purchase any real GPUs; all performance data is simulated. Once access to actual GPU hardware is available, only minimal adjustments will be needed to make this platform production-ready.

## 8 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/>