PROJECT PROPOSAL

Lottery Dapp

Decentralized Application

Instructor:

Mr. Yongchang He

Members:

Hai Nam Nguyen – 000520322 – nguyen0465@saskpolytech.ca

Cong Chi Tai Nguyen - 000516006 - nguyen6169@saskpolytech.ca

Xuan Hieu Nguyen – 000518043 – nguyen8191@saskpolytech.ca

Table of Contents

1. Problem Definition

Nowadays, it's necessary to maintain fairness and trust among participants in the Lottery. Thus, ensuring the integrity of lottery draws and protecting against **cyber-attacks, fraud, and manipulation** are top concerns. However, this system has been facing a huge amount of information technology challenges including **security, transaction handling, data privacy, and user experience**. Those challenges arise while handling transaction volumes leading to finding out the solution for **scalability, compliance,** mitigating the impact of **system failures** and ensuring **uninterrupted operations**. Addressing these challenges is important for lottery operators to keep the integrity, security, and reliability of our systems, enabling trust and confidence among participants.

2. Impact of Study

There are some reasons to explain why we should apply Blockchain for our Lottery Dapp:

- **Transparency**: Blockchain offers an unchangeable, transparent ledger of transactions. Every participant can inspect that the lottery procedure is fair and free from fraud or manipulation.
- **Decentralization**: traditional lotteries are frequently run by a centralized organization. Blockchain-based lotteries can work in a decentralized way, eliminating the requirement for a central authority and reducing corruption and manipulation.
- **Smart Contracts**: The lottery process from ticket sales to prize distribution can be run automatically using smart contracts, which are self-executed with the conditions of the agreement put into code. This will reduce administrative expenses and guarantee lottery execution fairness.

3. Contract Design

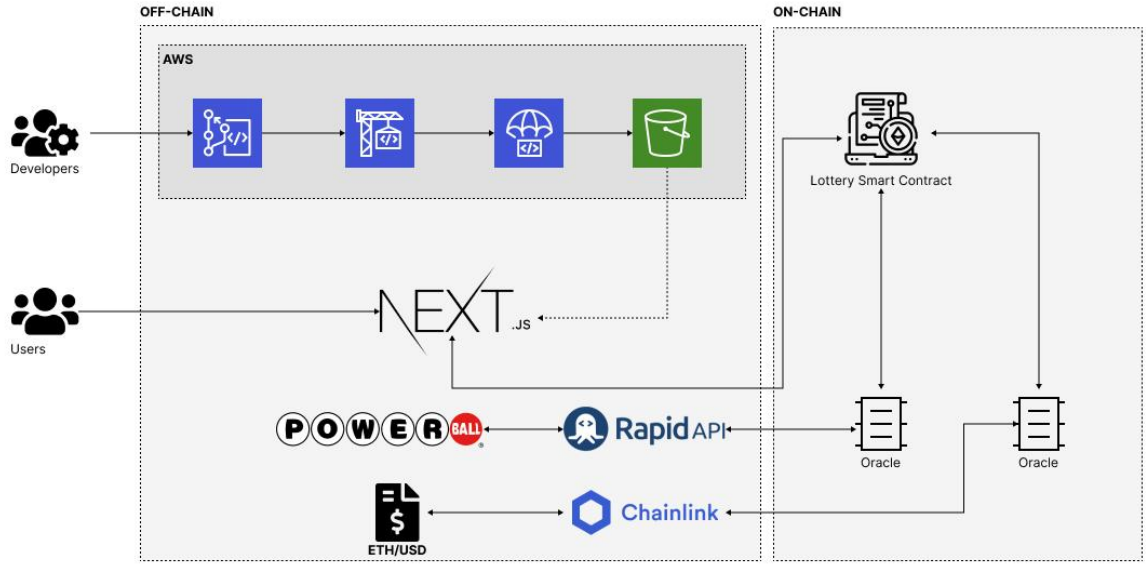
- Name of the contracts:
    - Lottery

    - DataConsumerV3
- Interfaces and libraries that the contracts will be use:
    - ILotteryDataInterface
    - IDataConsumerInterface
    - Chainlink Data Feeds
- The Lottery contract is designed to ensure a fair and transparent lottery system. It includes the following functions:
    - enter(): This a public function, meaning anyone can call it, and it is marked as payable to allow it to receive Ether along with the function call
    - random(): Define a private function and marked as view, which means it does not modify the contract state, and it returns a random number as a uint.
    - pickWinner(): This function is used to select a random winner from the list of players and then transfer the contract 80% balance to the winner
    - getPlayers(): This function allows anyone to view the list of players in the lottery
    - pickPowerballWinner(): This function is used to select a Powerball winner from the list of players and then transfer the contract 20% balance to the winner
    - modifier restricted(): This modifier restricts the access to pickWinner() and pickPowerballWinner() functions, allowing only the manager to call them
- DataConsumerV3 contract to connect our contract to asset pricing data for the ETH / USD feed, and Powerball result via RapidAPI by the Immediate-read pattern.
    - getEthToUsdRate(): Retrieves the latest Ethereum to USD exchange rate from the Chainlink Data Feed.
    - convertToUsd(): Converts an amount from Ether to USD based on the current exchange rate.
    - getPowerballResult(): Retrieves the latest Powerball result from RapidAPI.
- The contracts use the AggregatorV3Interface, an important library that enables communication with Chainlink Data Feeds. This interface ensures seamless integration with external price data, allowing the contract to obtain accurate exchange rates for USD conversion.

4. Contract Architecture

The Lottery application will have the architecture as in the figure below and use the latest technologies such as Amazon Web Services (AWS) for the DevOps, NextJS for the front-end application, and Smart Contract, Oracle on Ethereum blockchain for the logic application:

The Lottery application will use modern technologies like Amazon Web Services (AWS) or DevOps, NextJS for the front-end. It will also utilize Smart Contracts deployed on the Ethereum blockchain to manage the logic. Additionally, an Oracle will be integrated into the system to fetch external data and interact with the Smart Contracts on-chain.



For DevOps, the team will use AWS CodePipeline, CodeCommit, CodeBuild, and CodeDeploy to manage Continuous Integration and Continuous Delivery (CI/CD). This process is dedicated to the front-end application, which will be deployed to AWS Simple Storage Services (S3) and hosted as a static website.

The front-end application will be developed using the NextJS framework, which leverages the React library as its core. It will establish communication with the on-chain application logic through a smart contract named Lottery Smart Contract, storing crucial data from the front-

end application such as player addresses, randomly chosen numbers, and the list of players in the Lottery poll.

An on-chain Oracle will invoke APIs from Rapid API to fetch data from Powerball in the United States of America. This data will be securely stored on-chain and retrieved by the Lottery Smart Contract whenever the logic triggers to determine the winner, who will receive 20% of the total winning pool. Alongside the feature to fetch the price feed via Chain Link, users could know how much they win in real-time.

5. Conclusion

In conclusion, blockchain technology has several advantages that make it a desirable choice for establishing an equitable, open, and safe lottery system. Lottery organizers can use blockchain to build a decentralized, transparent, and unchangeable system that promotes participant trust, enables access to anywhere in the world, and uses smart contracts to automate procedures.