SPQR Quest - Ethereum-Based Republic

Lucius Junius Brutus

509 b.c.

Contents

1　Overview

SPQR Quest is a decentralized token-based game running on the Ethereum blockchain. The game integrates the ERC-20 standard, uses Chainlink VRF (Verifiable Random Function) for provably fair randomness, and leverages Uniswap for liquidity management. The token that powers the SPQR Quest ecosystem is **SPQR**.

SPQR Quest introduces a unique flip game where users can bet their SPQR tokens in hopes of winning more tokens based on a 50/50 chance generated by a Chainlink VRF random number generator. The ecosystem incorporates liquidity management, a fee structure, and essential features to protect the health of the token and game economy.

2　Features

2.1　ERC-20 Standard Compliant Token

The **SPQR** token is fully compliant with the ERC-20 token standard, which ensures compatibility with a wide range of wallets, exchanges, and decentralized applications (dApps).

2.2　Flip Game with Chainlink VRF Integration

Users can participate in a flip game where they bet a portion of their SPQR tokens. Chainlink VRF ensures fairness by generating a truly random number used to determine the outcome of each bet. A fee in ETH (called the **VRF Fee**) is required for each game to cover the cost of randomness provided by Chainlink.

2.3　Liquidity Management

Liquidity is automatically managed via Uniswap, ensuring the stability and tradability of SPQR tokens. The contract contains logic to add liquidity to Uniswap automatically, helping maintain healthy market conditions and reduce volatility.

2.4　Fee System

SPQR implements a dual-fee system for both buying and selling tokens:

- **Buy Fees**: A portion of every buy transaction is allocated towards liquidity and development.

- **Sell Fees**: Similar to buy fees, a percentage of each sell is directed toward liquidity and development.

The total buy and sell fees are capped to ensure that the ecosystem remains sustainable while preventing excessive costs for users.

2.5 Wallet and Transaction Limits

To maintain a healthy token ecosystem, **maximum transaction limits** and **maximum wallet size** are enforced. These limits can be adjusted by the owner but are designed to prevent single wallets from holding or transferring an excessive amount of the total supply. These limits help mitigate potential risks from large holders and ensure more decentralized token distribution.

2.6 Automated Market Maker (AMM) Protection

The contract allows specific addresses (e.g., liquidity pools) to be designated as **automated market makers**. This helps regulate trading behavior and apply different transaction rules to automated liquidity pools compared to individual users.

2.7 Swapping and Liquidity Management

The contract is equipped with an automated swapping mechanism where part of the collected fees are periodically swapped into ETH. This ETH is then used to add liquidity to Uniswap or sent to the development wallet for further project growth.

2.8 Exclusion from Fees and Limits

Specific addresses, such as those related to the project or essential infrastructure, can be excluded from transaction fees and wallet limits. This ensures smooth project operations without excessive restrictions.

2.9 Ownership and Control

The contract is owned by a central wallet, which allows control over critical functions, such as enabling trading, adjusting transaction limits, and managing fees. While the ownership model allows flexibility in adapting to market conditions, it also ensures accountability from the development team.

3 Technical Details

3.1 Contract Integrations

The SPQR Quest contract integrates with several decentralized protocols and technologies:

- **ERC-20**: Implements the standard ERC-20 interface, ensuring compatibility with exchanges, wallets, and DeFi protocols.

- **Uniswap Router**: The contract interacts with the Uniswap V2 Router for token swapping and liquidity provision.

- **Chainlink VRF**: The game uses Chainlink VRF for provably fair randomness, ensuring the integrity of the flip game's results.

- **ReentrancyGuard**: To protect against reentrancy attacks, the contract uses OpenZeppelin's ReentrancyGuard modifier.

3.2   Contract Parameters

- **Subscription ID**: A Chainlink VRF subscription ID is necessary for random number generation.

- **Key Hash**: A hash that identifies the VRF key pair used to generate randomness.

- **Callback Gas Limit**: Specifies the amount of gas provided for the callback function used to retrieve random numbers.

- **Transaction Limits**: The contract implements a maximum transaction amount and a maximum wallet size, both of which are adjustable by the owner.

3.3   Fee Structure

The fee structure ensures sustainable growth for the project while rewarding holders:

- **Buy Fees**:
  - Liquidity Fee: 0%
  - Development Fee: 3%

- **Sell Fees**:
  - Liquidity Fee: 4%
  - Development Fee: 3%

These fees are subject to a maximum cap of 20% on buys and 25% on sells to avoid excessive costs for traders. Additionally, liquidity is managed dynamically based on trading activity.

3.4   Liquidity and Tokenomics

- **Initial Supply**: 1,000,000 SPQR

- **Maximum Transaction Amount**: 5% of total supply

- **Maximum Wallet Size**: 5% of total supply

- **Automatic Liquidity Management**: A portion of transaction fees is swapped and added to Uniswap liquidity pools, ensuring token stability.

3.5 Chainlink VRF Integration

The flip game relies on Chainlink VRF, providing randomness that cannot be tampered with by any party. This ensures a provably fair game, where users can trust the outcome is not manipulated.

- **Flip Mechanism**: Users send a bet amount in SPQR tokens and pay a VRF Fee in ETH. The Chainlink VRF generates a random number to determine the outcome. If the outcome is favorable, the user wins back 199% of their bet amount.

3.6 Security and Safeguards

The contract is equipped with multiple security measures:

- **Reentrancy Protection**: Using the ReentrancyGuard ensures that users cannot manipulate the contract with recursive calls.

- **Limits on Maximum Transactions**: Helps prevent market manipulation or large dumps that could destabilize the token.

- **Emergency Functions**: Functions like `dangerClearCacheFlip()` allow users or admins to clear cached data in case of system issues, improving the resilience of the system.

3.7 Development Wallet

The **Development Wallet** holds funds from transaction fees to finance ongoing project development. It receives ETH from the contract's fee system and liquidity provisioning process.

4 Governance and Future Enhancements

SPQR Quest is governed by the contract owner, with certain functions allowing for adjustments as the game evolves. Features like fee adjustments, transaction limit changes, and liquidity management can be modified to ensure long-term sustainability.

In future updates, SPQR Quest may introduce:

- **Additional Game Mechanics**: Beyond the flip game, new randomness-based games can be added.

- **Community Governance**: Decentralized decision-making through community votes for specific updates or changes to the contract.

- **Cross-Chain Compatibility**: Expanding the game to other blockchain ecosystems.

5   Conclusion

SPQR Quest combines blockchain gaming, DeFi mechanics, and secure randomness through Chainlink VRF to create an engaging, fair, and transparent gaming experience. The use of Uniswap for liquidity management and the integration of a sustainable fee structure ensures a robust and healthy token ecosystem.

With its built-in safeguards, flexible governance, and exciting game mechanics, SPQR Quest aims to create a thriving community-driven environment in the Ethereum ecosystem.