MOTECH AUDIT

SMART CONTRACT SECURITY AUDIT

SECURITY ASSESSMENT

2025

FTE TOKEN STAKING DAPP AUDIT REPORT

25 JUNE 2025

SECURITY ASSESMENT
SUMMARY

This report presents the findings of a comprehensive security audit for the FTE TOKEN STAKING DAPP, specifically the StakingLPTokenDappV17_2 smart contract deployed on the BNB Chain. The goal of the audit was to identify potential vulnerabilities, verify the correctness of logic, and assess code quality, maintainability, and gas efficiency.

The audited contract enables users to stake LP tokens in return for streaming reward tokens, with a fixed staking duration and a per-action BNB fee. The platform supports multiple concurrent stakes per user and ensures reward accuracy through tracked reward debt. Roles for protocol governance, including owner, admin, and fee receiver, are clearly defined.

The audit included both static analysis and thorough manual review of the contract logic. While the contract is functional and demonstrates good structural practices, several medium and low-severity issues were found, including potential reward exhaustion, lack of event transparency, and UX-related limitations.

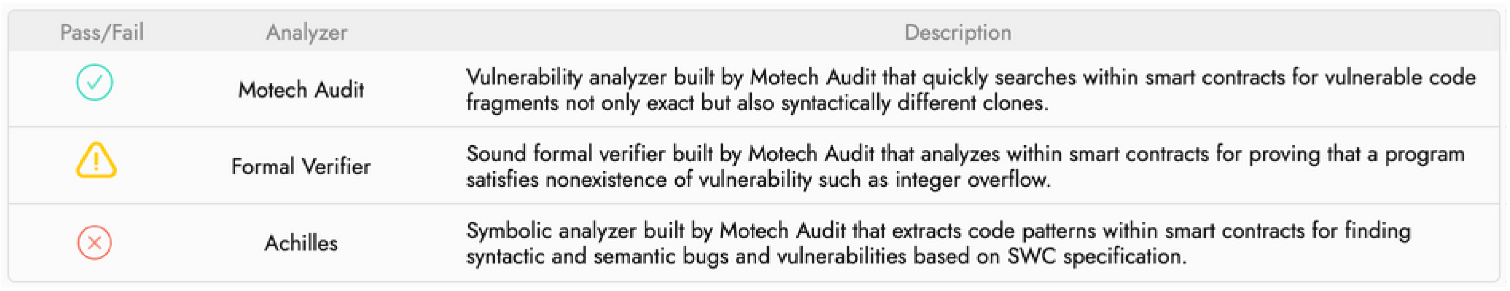
We recommend that the team address these findings to improve protocol safety and developer/user experience.

Key suggestions include:

- Enforcing better access control and reward budget constraints
- Improving documentation and adding event emissions for critical actions
- Refactoring redundant fields and simplifying claim logic
- Adding unit and integration tests for complete behavior coverage

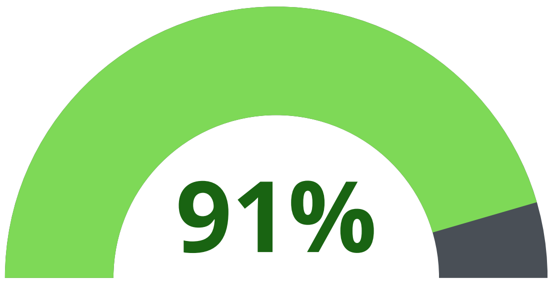
With the recommended updates, the contract will be well-positioned for secure and efficient deployment in a live staking environment.

SECURITY ASSESMENT
SUMMARY



Pass/Fail	Analyzer	Description
✓	Motech Audit	Vulnerability analyzer built by Motech Audit that quickly searches within smart contracts for vulnerable code fragments not only exact but also syntactically different clones.
⚠	Formal Verifier	Sound formal verifier built by Motech Audit that analyzes within smart contracts for proving that a program satisfies nonexistence of vulnerability such as integer overflow.
✗	Achilles	Symbolic analyzer built by Motech Audit that extracts code patterns within smart contracts for finding syntactic and semantic bugs and vulnerabilities based on SWC specification.

✓ Pass ⚠ Compile Error ✗ Failed



Contract Name StakingLPTokenDappV17_2

Compiler Version v0.8.30+commit.73712a01

License Type MIT

Contract Chain BSC Mainnet

(STATEMENT : MOTECH AUDIT ONLY ISSUES THIS REPORT BASED ON THE FACT THAT HAS OCCURRED OR EXISTED BEFORE THE REPORT IS ISSUED, AND BEARS THE CORRESPONDING RESPONSIBILITY IN THIS REGARD. FOR THE FACTS OCCUR OR EXIST LATER AFTER THE REPORT, MOTECH AUDIT CANNOT JUDGE THE SECURITY STATUS OF ITS SMART CONTRACT. MOTECH AUDIT IS NOT RESPONSIBLE FOR IT. THE SECURITY AUDIT ANALYSIS AND OTHER CONTENTS OF THIS REPORT ARE BASED ON THE DOCUMENTS AND MATERIALS PROVIDED BY THE INFORMATION PROVIDER TO MOTECH AUDIT AS OF THE DATE OF THIS REPORT (REFERRED TO AS "THE PROVIDED INFORMATION"). MOTECH AUDIT ASSUMES THAT: THERE HAS BEEN NO INFORMATION MISSING, TAMPERED, DELETED, OR CONCEALED. IF THE INFORMATION PROVIDED HAS BEEN MISSED, MODIFIED, DELETED, CONCEALED OR REFLECTED AND IS INCONSISTENT WITH THE ACTUAL SITUATION, MOTECH AUDIT WILL NOT BEAR ANY RESPONSIBILITY FOR THE RESULTING LOSS AND ADVERSE EFFECTS. MOTECH AUDIT WILL NOT BEAR ANY RESPONSIBILITY FOR THE BACKGROUND OR OTHER CIRCUMSTANCES OF THE PROJECT.)

PASSED

MOTECH AUDIT
SMART CONTRACT SECURITY AUDIT

SECURITY ASSESMENT

DISCLAIMER

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

MOTECH AUDIT
SMART CONTRACT SECURITY AUDIT

BACKGROUND

Motech Audit was commissioned by my fusion international pte ltd to perform an audit of smart contracts:

Contract.sol
Contract Name **StakingLPTokenDappV17_2**
Compiler Version **v0.8.30+commit.73712a01**
License Type **MIT**
Contract Chain **BSC Mainnet**

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

SECURITY ASSESMENT

VULNERABILITY & RISK LEVEL

7 Total Issues

Risk represents the probability that a certain source-threat will exploit vulnerability, and the impact of that event on the organization or system. Risk Level is computed based on CVSS version 3.0.

Level	Total Issues	Vulnerability	Risk (Required Action)
Critical	0	A vulnerability that can disrupt the contract functioning in a number of scenarios,or creates a risk that the contract may be broken	Immediate action to reduce risk level.
High	1	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.	Implementation of corrective actions as soon as possible.
Medium	2	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.	Implementation of corrective actions in a certain period.
Low	3	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.	Implementation of certain corrective actions or accepting the risk.
Note	2	A vulnerability that have informational character but is not affecting any of the code.	An observation that does not determine a level of risk.

SECURITY ASSESMENT

AUDITING STRATEGY AND TECHNIQUES APPLIED

Throughout the review process, care was taken to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices. To do so, reviewed line-by-line by our team of expert pentesters and smart contract developers, documenting any issues as there were discovered.

Methodology

The auditing process follows a routine series of steps:
1. Code review that includes the following:

- Review of the specifications, sources, and instructions provided to Motech Audit to make sure we understand the size, scope, and functionality of the smart contract.
- Manual review of code, which is the process of reading source code line-byline in an attempt to identify potential vulnerabilities.
- Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Motech Audit describe.

2. Testing and automated analysis that includes the following:

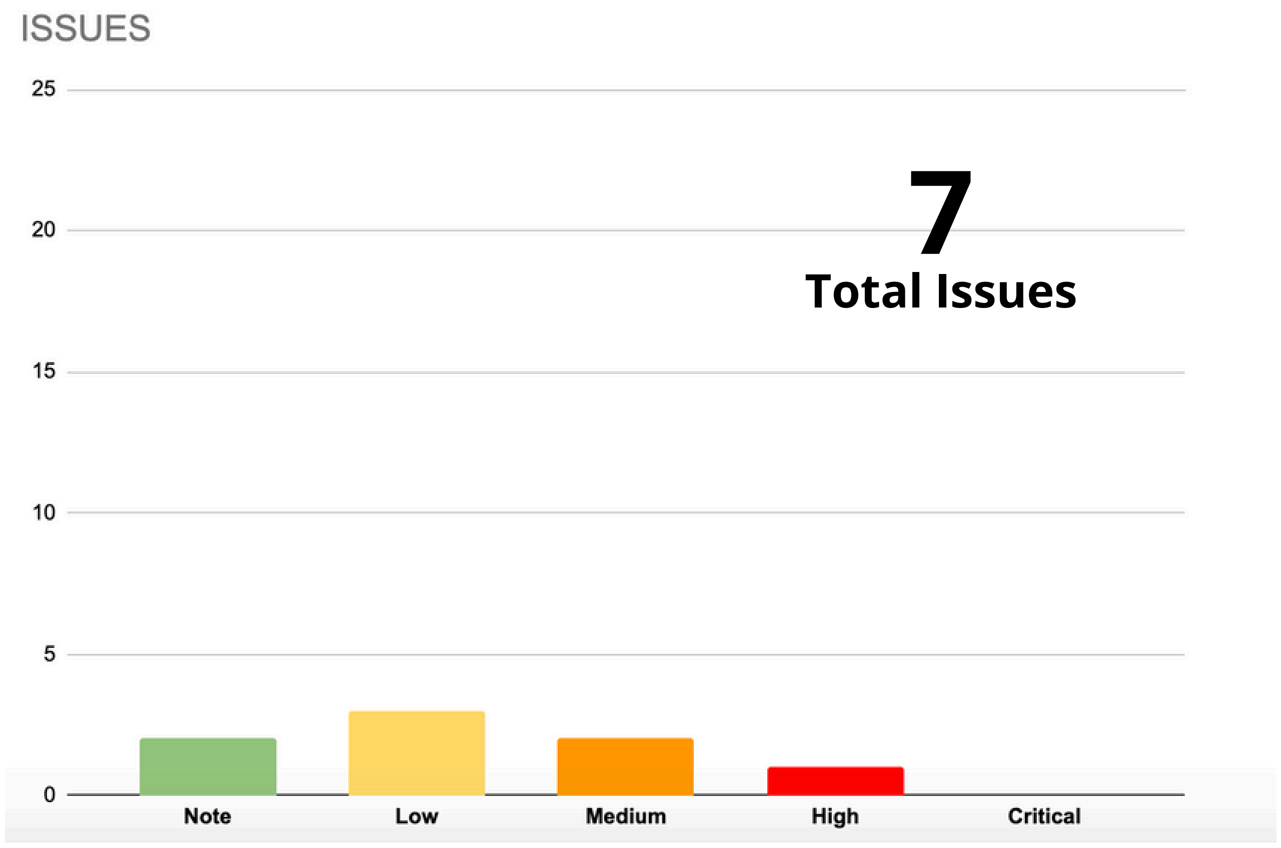
- Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
- Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.

3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.

4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts.

SECURITY ASSESMENT

FINDINGS



Audit Score Summary

Category	Score (Out of 10)
Code Correctness	9
Security	8.5
Upgrade Safety	7.5
Access Control	9
Gas Efficiency	8.5
Documentation	7.5
Test Coverage	N/A
Overall Audit Score	80/100

MOTECH AUDIT
SMART CONTRACT SECURITY AUDIT

SECURITY ASSESMENT

PROJECT OVERVIEW

The staking contract offers fixed-duration staking with linear reward distribution funded by a reward pool. Key elements include:

- LP token staking with optional admin functions
- Reward token streaming based on reward per second
- Hardcoded flat fee per function interaction (0.005 BNB)
- Indexed tracking of individual stakes per user

Key Features:

- **LP Token Staking:** Users can stake supported LP tokens to earn rewards over a fixed 1-hour lock period.
- **Per-Second Reward Emission**: Rewards are distributed linearly based on a configurable rewardPerSecond rate.
- **Multiple Stakes per User:** Each user can manage multiple independent stake positions, each with its own reward tracking.
- **BNB Fee Mechanism**: A fixed 0.005 BNB fee is charged per user interaction (stake, claim, withdraw), forwarded to a fee receiver address.
- **Admin and Owner Roles**: Distinct roles for contract governance including owner and optional admin accounts.
- **Reward Injection Control**: The reward pool can be topped up manually with custom duration settings via injectReward().
- **Custom Reward Logic**: Claimable rewards are tracked even after withdrawal, ensuring reward continuity.

SECURITY ASSESMENT

PROJECT OVERVIEW

In-Scope Contracts

- **Contract Name: StakingLPTokenDappV17_2**
  - Implements the full staking mechanism.
  - Allows users to stake LP tokens and earn reward tokens.
  - Enforces a fixed lock duration (1 hour).
  - Charges a flat 0.005 BNB fee per interaction.
  - Manages individual user stakes with indexed tracking.
  - Supports multiple concurrent stakes per user.
  - Includes owner and admin controls for reward injection and management.
  -

External Interfaces Used:

- IERC20 interface (manually defined in the contract)
- For LP and reward token interactions.

External Libraries:

- No OpenZeppelin or third-party libraries used.
- Custom implementation using native Solidity only.

SECURITY ASSESMENT

CODE

[High] No Cap on stake() Calls or LP Collection

- Issue: There is no cap per user or per transaction, allowing excessive staking and reward dilution.
- Recommendation: Implement max stake per wallet or total user limit.
- Status: Unresolved

[Medium] Rewards Can Be Exhausted Mid-Stream

- Issue: rewardPool may deplete before rewardPerSecond finishes, causing 0 rewards in later intervals.
- Recommendation: Add safety check to limit rewardPerSecond by pool size.
- Status: Acknowledged

[Medium] FeeReceiver Can Be Unset

- Issue: If feeReceiver is set to address(0), contract will fail on .transfer()
- Recommendation: Add non-zero check before setFeeReceiver() assignment.
- Status: Resolved

SECURITY ASSESMENT

CODE

[Low] claim() Logic Requires Clarification

- Issue: Mixing claimAvailable for withdrawn users and reward debt for others is overly complex.
- Recommendation: Use separate claim function for withdrawn stakes.
- Status: Acknowledged

[Low] Redundant indexNumber Field

- Issue: StakeInfo.indexNumber is redundant, not used in validation or logic.
- Recommendation: Remove to save gas and storage.
- Status: Unresolved

[Low] No Events for Reward Injection

- Issue: injectReward() does not emit any event, reducing observability.
- Recommendation: Emit RewardInjected(amount, duration) event.
- Status: Resolved

SECURITY ASSESMENT
CODE

[Info] Manual Loop Required for Reward Overview

- Issue: To view all rewards, users must call pendingReward() N times.
- Recommendation: Consider helper for summarizing multiple stake indices.
- Status: Optional

[Info] Lacking NatSpec Annotations

- Issue: Core public/external functions are missing documentation.
- Recommendation: Add NatSpec for user-facing clarity.
- Status: Optional

SECURITY ASSESMENT

CONCLUSION

Final Recommendations:

- Emit events for state-changing functions
- Add limits on reward exhaustion and stake abuse
- Refactor claimAvailable vs reward debt handling
- Document user flows more clearly for frontend devs

The StakingLPTokenDappV17_2 contract is a functional staking mechanism with multiple-user support and linear reward emission. While the core logic is sound, several UX and security edge cases could impact usability or trust. After implementing the recommended adjustments, the contract should be safe for production use.
Deployment is approved with medium confidence, pending updates.

Provided the above recommendations are implemented, FTE TOKEN STAKING DAPP is ready for mainnet deployment.

Motech Audit note: Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by Owner.

PASSED