CFG NINJA AUDITS

Security Assessment

Flexvis Insured Investment

March 9, 2023

Audit Status: Pass

Audit Edition: Advance

Table of Contents

Assessment Summary

This report has been prepared for Flexvis Insured Investment on the Binance Smart Chain network. CFGNINJA provides both client-centered and user-centered examination of the smart contracts and their current status when applicable. This report represents the security assessment made to find issues and vulnerabilities on the source code along with the current liquidity and token holder statistics of the protocol.

A comprehensive examination has been performed, utilizing Cross Referencing, Static Analysis, In-House Security Tools, and line-by-line Manual Review.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Inspecting liquidity and holders statistics to inform the current status to both users and client when applicable.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Verifying contract functions that allow trusted and/or untrusted actors to mint, lock, pause, and transfer assets.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders
- Thorough line-by-line manual review of the entire codebase by industry experts.

Project Overview

Token Summary

Parameter	Result
Address	0xb2eF98BB03a26fc7b46BEa6018800b654F932275
Name	Flexvis
Token Tracker	Flexvis (FLX)
Decimals	
Supply	
Platform	Binance Smart Chain
compiler	v0.8.9+commit.e5eed63a
Contract Name	InsuranceInvestment
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	https://bscscan.com/address/0xb2eF98BB03a26fc7b46BEa6018800b654F932275#code
Payment Tx	Corporate

Main Contract Assessed Contract Name

Name	Contract	Live
Flexvis	0xb2eF98BB03a26fc7b46BEa6018800b654F932275	No

TestNet Contract Assessed Contract Name

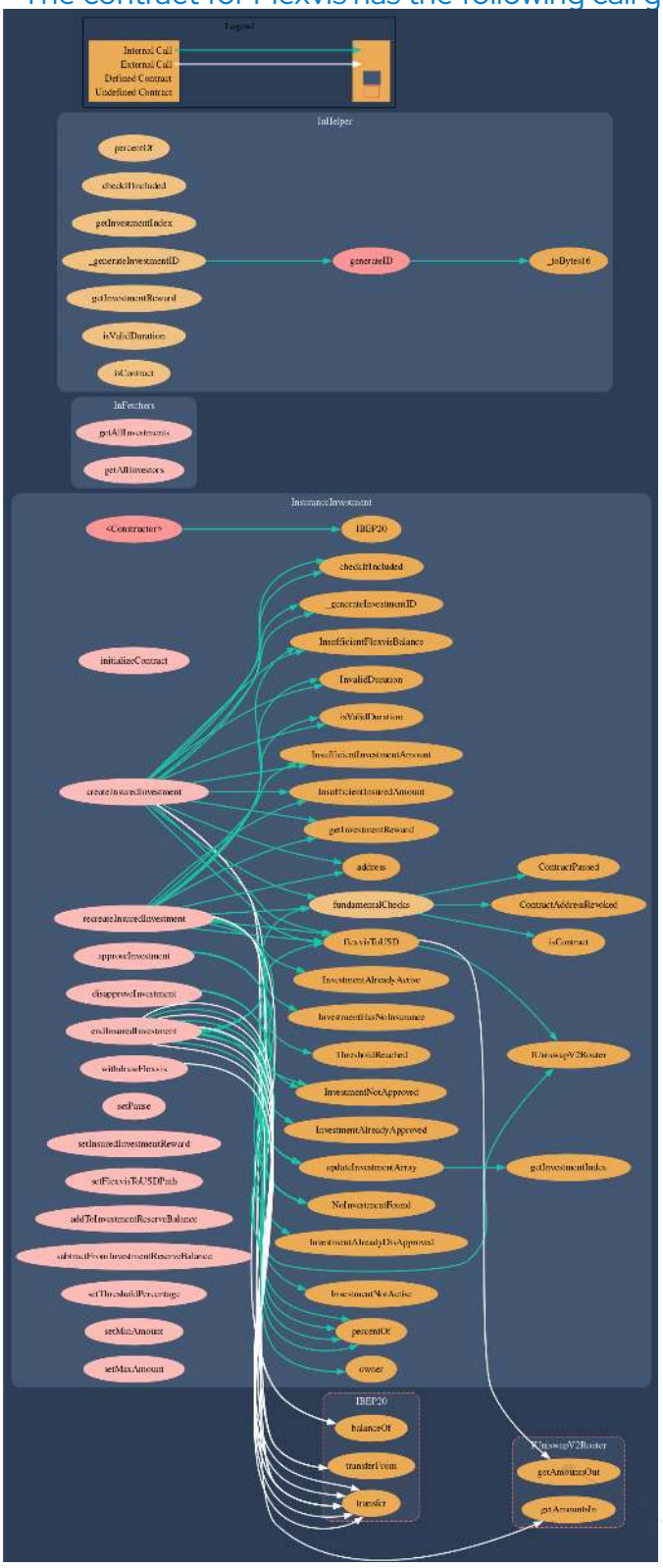
Name	Contract	Live
Flexvis	0x222fd1461b745793f330780124203e33c3cd2771	No

Solidity Code Provided

SolID	File Sha-1	FileName
InsuranceInvestment	72efdc3b25f2f161428f9f5fe7ab8d70da84af35	InsuranceInvestment.sol
InsuranceInvestment	cbf743ad051714c05238c12b4a09e4e379a44b83	InFetchers.sol
InsuranceInvestment	6143dc6b798bbbefe2969cefb79c7e73fa9a6674	InHelper.sol

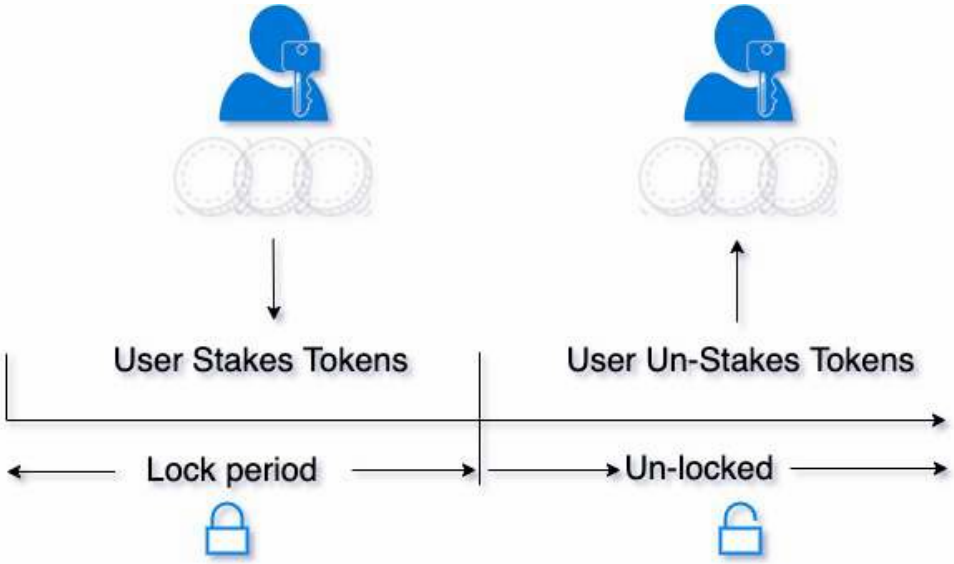
Call Graph

The contract for Flexvis has the following call graph structure.



What is a Staking Contract

A smart contract which allows users to stake and un-stake a specified ERC20 token. Staked tokens are locked for a specific length of time (set by the contrat owner at the outset). Once the time period has elapsed, the user can remove their tokens again.

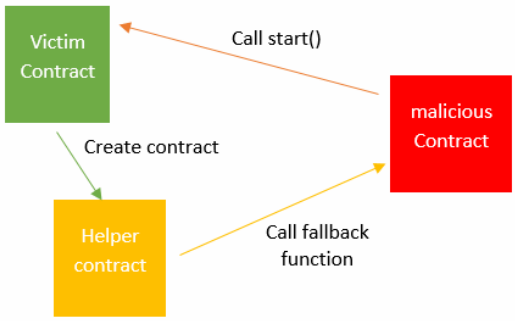


Reentrancy Check

The Project Owners of Flexvis have implemented Reentrancy Guard Library

The Team has done a great job to avoid potential reentrancy issues in the contract.

You can read more about the reentrancy library used.
ReentrancyGuard



KYC Information

The Project Owners of Flexvis is not KYC.

KYC Information Notes:

Auditor Notes:

Project Owner Notes:

Smart Contract Vulnerability Checks

The Smart Contract Weakness Classification Registry (SWC Registry) is an implementation of the weakness classification scheme proposed in EIP-1470. It is loosely aligned to the terminologies and structure used in the Common Weakness Enumeration (CWE) while overlaying a wide range of weakness variants that are specific to smart contracts.

ID	Severity	Name	File	location
SWC-100	Pass	Function Default Visibility	InsuranceInvestment.sol	L: 0 C: 0
SWC-101	Pass	Integer Overflow and Underflow.	InsuranceInvestment.sol	L: 0 C: 0
SWC-102	Pass	Outdated Compiler Version file.	InsuranceInvestment.sol	L: 0 C: 0
SWC-103	Pass	A floating pragma is set.	InsuranceInvestment.sol	L: 2 C: 3
SWC-104	Pass	Unchecked Call Return Value.	InsuranceInvestment.sol	L: 0 C: 0
SWC-105	Pass	Unprotected Ether Withdrawal.	InsuranceInvestment.sol	L: 0 C: 0
SWC-106	Pass	Unprotected SELFDESTRUCT Instruction	InsuranceInvestment.sol	L: 0 C: 0
SWC-107	Pass	Read of persistent state following external call.	InsuranceInvestment.sol	L: 0 C: 0
SWC-108	Pass	State variable visibility is not set..	InsuranceInvestment.sol	L: 0 C: 0
SWC-109	Pass	Uninitialized Storage Pointer.	InsuranceInvestment.sol	L: 0 C: 0
SWC-110	Pass	Assert Violation.	InsuranceInvestment.sol	L: 0 C: 0

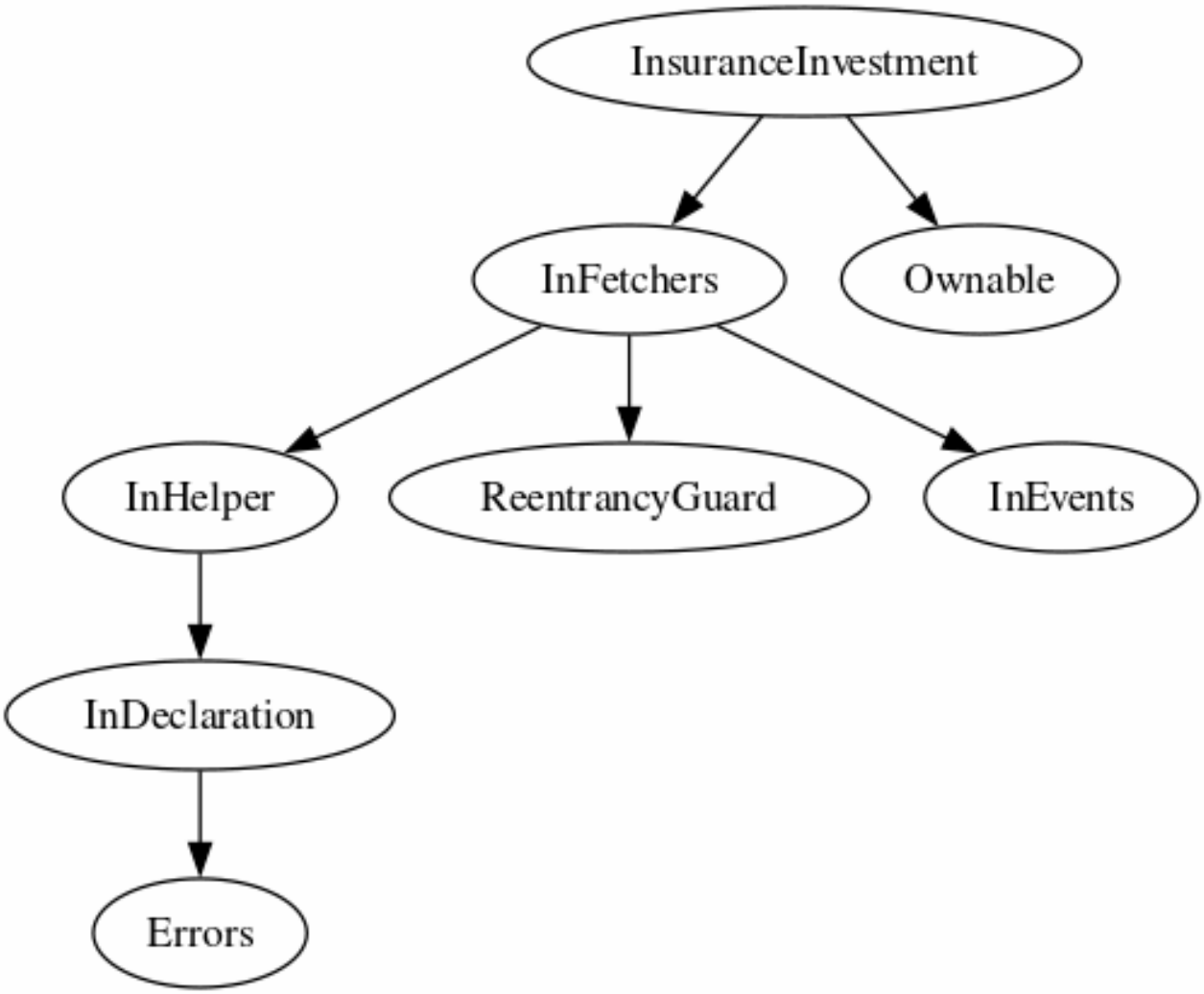
ID	Severity	Name	File	location
SWC-111	Pass	Use of Deprecated Solidity Functions.	InsuranceInvestment.sol	L: 0 C: 0
SWC-112	Pass	Delegate Call to Untrusted Callee.	InsuranceInvestment.sol	L: 0 C: 0
SWC-113	Pass	Multiple calls are executed in the same transaction.	InsuranceInvestment.sol	L: 0 C: 0
SWC-114	Pass	Transaction Order Dependence.	InsuranceInvestment.sol	L: 0 C: 0
SWC-115	Pass	Authorization through tx.origin.	InsuranceInvestment.sol	L: 0 C: 0
SWC-116	Pass	A control flow decision is made based on The block.timestamp environment variable.	InsuranceInvestment.sol	L: 0 C: 0
SWC-117	Pass	Signature Malleability.	InsuranceInvestment.sol	L: 0 C: 0
SWC-118	Pass	Incorrect Constructor Name.	InsuranceInvestment.sol	L: 0 C: 0
SWC-119	Pass	Shadowing State Variables.	InsuranceInvestment.sol	L: 0 C: 0
SWC-120	Pass	Potential use of block.number as source of randonmness.	InsuranceInvestment.sol	L: 0 C: 0
SWC-121	Pass	Missing Protection against Signature Replay Attacks.	InsuranceInvestment.sol	L: 0 C: 0
SWC-122	Pass	Lack of Proper Signature Verification.	InsuranceInvestment.sol	L: 0 C: 0
SWC-123	Pass	Requirement Violation.	InsuranceInvestment.sol	L: 0 C: 0
SWC-124	Pass	Write to Arbitrary Storage Location.	InsuranceInvestment.sol	L: 0 C: 0

ID	Severity	Name	File	location
SWC-125	Pass	Incorrect Inheritance Order.	InsuranceInvestment.sol	L: 0 C: 0
SWC-126	Pass	Insufficient Gas Griefing.	InsuranceInvestment.sol	L: 0 C: 0
SWC-127	Pass	Arbitrary Jump with Function Type Variable.	InsuranceInvestment.sol	L: 0 C: 0
SWC-128	Pass	DoS With Block Gas Limit.	InsuranceInvestment.sol	L: 0 C: 0
SWC-129	Pass	Typographical Error.	InsuranceInvestment.sol	L: 0 C: 0
SWC-130	Pass	Right-To-Left-Override control character (U +202E).	InsuranceInvestment.sol	L: 0 C: 0
SWC-131	Pass	Presence of unused variables.	InsuranceInvestment.sol	L: 0 C: 0
SWC-132	Pass	Unexpected Ether balance.	InsuranceInvestment.sol	L: 0 C: 0
SWC-133	Pass	Hash Collisions with Multiple Variable Length Arguments.	InsuranceInvestment.sol	L: 0 C: 0
SWC-134	Pass	Message call with hardcoded gas amount.	InsuranceInvestment.sol	L: 0 C: 0
SWC-135	Pass	Code With No Effects (Irrelevant/Dead Code).	InsuranceInvestment.sol	L: 0 C: 0
SWC-136	Pass	Unencrypted Private Data On-Chain.	InsuranceInvestment.sol	L: 0 C: 0

We scan the contract for additional security issues using MYTHX and industry-standard security scanning tools.

Inheritance

The contract for Flexvis has the following inheritance structure.



Privileged Functions (onlyOwner)

Please Note if the contract is Renounced none of this functions can be executed.

Function Name	Parameters	Visibility
renounceOwnership		public
transferOwnership	newOwner (address)	public
setMaxAmount	uint256 _amount	external
setMinAmount	uint256 _amount	external
setThresholdPercentage	uint16 percentage	external
subtractFromInvestmentReserveBalance	uint256 amount	external
addToInvestmentReserveBalance	uint256 amount	external
setFlexvisToUSDPath	address[] memory _flexvisToUSD	external
setInsuredInvestmentReward	uint256[] memory percentages,uint256[] memory durations	external
setPause	bool to	external

Function Name	Parameters	Visibility
withdrawFlexvis	uint256 amount	external
disapproveInvestment	bytes16 investmentID, address investmentOwner	external
approveInvestment	bytes16 investmentID, address investmentOwner	external
createInsuredInvestment	uint256 amount,uint256 duration,uint256 insuredAmount	external
initializeContract	address _treasury,address _flexvis,address _busd,address _wbnb,address _router,address _investment	external

Smart Contract Advance Checks

ID	Severity	Name	Result	Status
FLX-01	Minor	Potential Sandwich Attacks.	Pass	Not-Found
FLX-02	Minor	Function Visibility Optimization	Pass	Resolved
FLX-03	Minor	Lack of Input Validation.	Pass	Not-Found
FLX-04	Major	Centralized Risk In addLiquidity.	Pass	Not-Found
FLX-05	Major	Missing Event Emission.	Pass	Resolved
FLX-06	Minor	Conformance with Solidity Naming Conventions.	Pass	Not Found
FLX-07	Minor	State Variables could be Declared Constant.	Pass	Resolved
FLX-08	Major	Dead Code Elimination.	Pass	Not-Found
FLX-09	Major	Third Party Dependencies.	Pass	Not Found
FLX-10	Major	Initial Token Distribution.	Pass	Not-Found
FLX-11	Critical	Initialization don't validate parameters.	Pass	Resolved
FLX-12	Major	Centralization Risks In The X Role	Pass	Not Found
FLX-13	Informational	Extra Gas Cost For User..	Pass	Not-Found
FLX-14	Medium	Unnecessary Use Of SafeMath	Pass	Not-Found
FLX-15	Medium	Symbol Length Limitation due to Solidity Naming Standards.	Pass	Not-Found

ID	Severity	Name	Result	Status
FLX-16	Medium	Invalid collection of Taxes during Transfer.	Pass	Not-Found

Technical Findings Summary

Classification of Risk

Severity	Description
Critical	Risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
Major	Risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
Medium	Risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform
Minor	Risks can be any of the above but on a smaller scale. They generally do not compromise the overall integrity of the Project, but they may be less efficient than other solutions.
Informational	Errors are often recommended to improve the code's style or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

Findings

Severity	Found	Pending	Resolved
Critical	0	0	0
Major	0	0	0
Medium	0	0	0
Minor	0	0	0
Informational	0	0	0
Total	0	0	0

Social Media Checks

Social Media	URL	Result
Twitter	https://twitter.com/flexvis	Pass
Other	https://www.instagram.com/flexvis/	Pass
Website	https://www.flexvis.io/	Pass
Telegram	https://t.me/flexvis	Pass

We recommend to have 3 or more social media sources including a completed working websites.

Social Media Information Notes:

Auditor Notes: undefined

Project Owner Notes:

Aduit Result

Final Audit Score

Review	Score
Security Score	90
Auditor Score	95

The Following Score System Has been Added to this page to help understand the value of the audit, the maximun score is 100, however to attain that value the project most pass and provide all the data needed for the assessment. Our Passing Score has been changed to 80 Points, if a project does not attain 80% is an automatic failure. Read our notes and final assessment below.

Audit Passed

Assessment Results

Important Notes:

- No vulnerabilities found.
- Code has Reentrancy Library to protect code.
- Code logic is great and we'll documented.

Auditor Score =95
Audit Passed

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that actagainst the nature of decentralization, such as explicit ownership or specialized access roles incombination with a mechanism to relocate funds.

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimalEVM opcodes resulting in a reduction on the total gas cost of a transaction.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on howblock.timestamp works.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functionsbeing invoke-able by anyone under certain circumstances.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that mayresult in a vulnerability.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to makethe codebase more legible and, as a result, easily maintainable.

Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code,such as a constructor assignment imposing different require statements on the input variables than a setterfunction.

Coding Best Practices

ERC 20 Conding Standards are a set of rules that each developer should follow to ensure the code meet a set of creterias and is readable by all the developers.

Disclaimer

CFGNINJA has conducted an independent security assessment to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the reviewed code for the scope of this assessment. This report does not constitute agreement, acceptance, or advocation for the Project, and users relying on this report should not consider this as having any merit for financial advice in any shape, form, or nature. The contracts audited do not account for any economic developments that the Project in question may pursue, and the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude, and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are entirely free of exploits, bugs, vulnerabilities or deprecation of technologies.

All information provided in this report does not constitute financial or investment advice, nor should it be used to signal that any persons reading this report should invest their funds without sufficient individual due diligence, regardless of the findings presented. Information is provided 'as is, and CFGNINJA is under no covenant to audited completeness, accuracy, or solidity of the contracts. In no event will CFGNINJA or its partners, employees, agents, or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions or actions with regards to the information provided in this audit report.

The assessment services provided by CFGNINJA are subject to dependencies and are under continuing development. You agree that your access or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies with high levels of technical risk and uncertainty. The assessment reports could include false positives, negatives, and unpredictable results. The services may access, and depend upon, multiple layers of third parties.