FusionTech

GREENWAVES

Audit Report

www.greenwav.es

Table of Contents

1 | Audit Summary

Project Name	GREENWAVES - (https://greenwav.es/)
Platform	BINANCE SMART CHAIN
Language	Solidity
Contract Address	0x73Fed59E116AE579D9D6C3796D66ACFec39C7AC5

Delivery Date	March 20, 2024
Audit Methodology	Static Analysis, Manual Review
Key Components	GreenWaves

Vulnerability Summary

Vulnerability Summary	Total	ⓘ Pending	ⓘ Declined	ⓘ Acknowledged	ⓘ Partially Resolved	ⓘ Resolved
Critical	0	0	0	0	0	0
Major	0	0	0	0	0	0
Medium	0	0	0	0	0	0
Minor	0	0	0	0	0	0
Informational	0	0	0	0	0	0
Discussion	0	0	0	0	0	0

1.1 | Audit Scope

External Dependencies

This audit focused on identifying security flaws in code and the design of EscrowDapp Contract. It was conducted on the source code provided by the EscrowDapp team. The following files were made available in the course of the review:

Privileged Functions

The contract contains the following privileged functions that are restricted by role with the modifier. Since the contract is the owner cannot modify the contract configurations and address attributes.

Audit methodology

Dependencies

Our security audit process for smart contract includes two steps:

• Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using public and RK87, our in-house smart contract security analysis tool.

• Manual audit of the codes for security issues. The contracts are manually analyzed to look for any potential problems.

1.3 | Source Code

```
interface UniSwapFactory {
   event PairCreated(address indexed token0, address indexed token1, address pair, uint);

   function feeTo() external view returns (address);

   function feeToSetter() external view returns (address);

    function getPair(address tokenA, address tokenB) external view returns (address pair);

    function allPairs(uint) external view returns (address pair);

   function allPairsLength() external view returns (uint);

     function createPair(address tokenA, address tokenB) external returns (address pair);

   function setFeeTo(address) external;

   function setFeeToSetter(address) external;
}
```

```
interface IIUniSwapPair {
   event Approval(address indexed owner, address indexed spender, uint value);
   event Transfer(address indexed from, address indexed to, uint value);

   function name() external pure returns (string memory);

   function symbol() external pure returns (string memory);

   function decimals() external pure returns (uint8);

   function totalSupply() external view returns (uint);

   function balanceOf(address owner) external view returns (uint);

    function allowance(address owner, address spender) external view returns (uint);

   function approve(address spender, uint value) external returns (bool);

   function transfer(address to, uint value) external returns (bool);

    function transferFrom(address from, address to, uint value) external returns (bool);

   function DOMAIN_SEPARATOR() external view returns (bytes32);

   function PERMIT_TYPEHASH() external pure returns (bytes32);

   function nonces(address owner) external view returns (uint);

    function permit(address owner, address spender, uint value, uint deadline, uint8 v, bytes32 r, bytes32 s) external;
```

```
event Sync(uint112 reserve0, uint112 reserve1);

  function MINIMUM_LIQUIDITY() external pure returns (uint);

  function factory() external view returns (address);

  function token0() external view returns (address);

  function token1() external view returns (address);

   function getReserves() external view returns (uint112 reserve0, uint112 reserve1, uint32 blockTimestampLast);

  function price0CumulativeLast() external view returns (uint);

  function price1CumulativeLast() external view returns (uint);

  function kLast() external view returns (uint);

  function mint(address to) external returns (uint liquidity);

  function burn(address to) external returns (uint amount0, uint amount1);

   function swap(uint amount0Out, uint amount1Out, address to, bytes calldata data) external;

  function skim(address to) external;

  function sync() external;

  function initialize(address, address) external;
}
```

```
function swapExactETHForTokens(uint amountOutMin, address[] calldata path, address to, uint deadline)
  external
  payable
  returns (uint[] memory amounts);

   function swapTokensForExactETH(uint amountOut, uint amountInMax, address[] calldata path, address to, uint deadline)
  external
  returns (uint[] memory amounts);

   function swapExactTokensForETH(uint amountIn, uint amountOutMin, address[] calldata path, address to, uint deadline)
  external
  returns (uint[] memory amounts);

   function swapETHForExactTokens(uint amountOut, address[] calldata path, address to, uint deadline)
  external
  payable
  returns (uint[] memory amounts);

    function quote(uint amountA, uint reserveA, uint reserveB) external pure returns (uint amountB);

   function getAmountOut(uint amountIn, uint reserveIn, uint reserveOut) external pure returns (uint amountOut);

   function getAmountIn(uint amountOut, uint reserveIn, uint reserveOut) external pure returns (uint amountIn);

   function getAmountsOut(uint amountIn, address[] calldata path) external view returns (uint[] memory amounts);

   function getAmountsIn(uint amountOut, address[] calldata path) external view returns (uint[] memory amounts);
}
```

```
contract DxFeeToken is Context, IERC20, Ownable {
  using SafeMath for uint256;
  address private dead = 0x000000000000000000000000000000000000dEaD;
  uint256 public maxLiqFee = 10;
  uint256 public maxTaxFee = 10;
  uint256 public maxDevFee = 10;
  uint256 public minMxTxPercentage = 50;
  uint256 public maxSellTaxFee = 20;
  uint256 public maxSellLiqFee = 20;

  mapping (address => uint256) private _rOwned;
  mapping (address => uint256) private _tOwned;
    mapping (address => mapping (address => uint256)) private
_allowances;
  mapping (address => bool) private _isExcludedFromFee;
  mapping (address => bool) private _isExcluded;
  mapping (address => bool) private _isdevWallet;

  address[] private _excluded;
  address public _devWalletAddress;  // team wallet here
  address public router;
  address public basePair;
  uint256 private constant MAX = ~uint256(0);
  uint256 private _tTotal;
  uint256 private _rTotal;
  uint256 private _tFeeTotal;
  bool public mintedByDxsale = true;
  string private _name;
  string private _symbol;
  uint8 private _decimals;
```

```
constructor (address tokenOwner,string memory name_, string memory
symbol_,uint8 decimal_, uint256 amountOfTokenWei,uint8[5] memory setFees,
uint256[6] memory maxFees, address devWalletAddress_, address _router,
address _basePair) {
    _name = name_;
    _symbol = symbol_;
    _decimals = decimal_;
    _tTotal = amountOfTokenWei;
    _rTotal = (MAX - (MAX % _tTotal));
    router = _router;
    basePair = _basePair;
    _rOwned[tokenOwner] = _rTotal;

    maxTaxFee = maxFees[0];
    maxLiqFee = maxFees[1];
    maxDevFee = maxFees[2];
    minMxTxPercentage = maxFees[3];
    maxSellTaxFee = maxFees[4];
    maxSellLiqFee = maxFees[5];
    _taxFee = setFees[0];
    _previousTaxFee = _taxFee;
    _liquidityFee = setFees[1];
    _previousLiquidityFee = _liquidityFee;
    _devFee = setFees[2];
    _previousDevFee = _devFee;
    _sellTaxFee = setFees[3];
    _previousSellTaxFee = _sellTaxFee;
    _sellLiqFee = setFees[4];
    _previousSellLiqFee = _sellLiqFee;
    _devWalletAddress = devWalletAddress_;

    _maxTxAmount = amountOfTokenWei;
    numTokensSellToAddToLiquidity = amountOfTokenWei.mul(1).div(1000); //0.1%
    swapAndLiquifyEnabled = true;
```

```
function addLiquidity(uint256 tokenAmount, uint256 ETHAmount) private {
    // approve token transfer to cover all possible scenarios
    _approve(address(this), address(uniswapV2Router), tokenAmount);

    // add the liquidity

    try uniswapV2Router.addLiquidityETH{value : ETHAmount}(
      address(this),
      tokenAmount,
      0, // slippage is unavoidable
      0, // slippage is unavoidable
      dead,
      block.timestamp
    ) {

    }

    catch (bytes memory) {
      try uniswapV2Router.addLiquidityBNB{value : ETHAmount}(
        address(this),
        tokenAmount,
        0, // slippage is unavoidable
        0, // slippage is unavoidable
        dead,
        block.timestamp
      ) {

      }
      catch (bytes memory) {
        try uniswapV2Router.addLiquidityAVAX{value : ETHAmount}(
          address(this),
          tokenAmount,
          0, // slippage is unavoidable
          0, // slippage is unavoidable
          dead,
          block.timestamp
        ) {

        }
```

```
catch (bytes memory) {
try uniswapV2Router.addLiquidityHT{value : ETHAmount}(
address(this),
tokenAmount,
0, // slippage is unavoidable
0, // slippage is unavoidable
dead,
block.timestamp
) {

}
catch (bytes memory) {

uniswapV2Router.addLiquidityETH{value : ETHAmount}(
address(this),
tokenAmount,
0, // slippage is unavoidable
0, // slippage is unavoidable
dead,
block.timestamp
);
}//this method is responsible for taking all fee, if takeFee is true
 function _tokenTransfer(address sender, address recipient, uint256 amount,bool takeFee) private {
   if(!takeFee)
     removeAllFee();

   if (_isExcluded[sender] && !_isExcluded[recipient]) {
     _transferFromExcluded(sender, recipient, amount);
   } else if (!_isExcluded[sender] && _isExcluded[recipient]) {
     _transferToExcluded(sender, recipient, amount);
   } else if (!_isExcluded[sender] && !_isExcluded[recipient]) {
     _transferStandard(sender, recipient, amount);
   } else if (_isExcluded[sender] && _isExcluded[recipient]) {
     _transferBothExcluded(sender, recipient, amount);
   } else {
     _transferStandard(sender, recipient, amount);
   }

 }
```

2 | Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimerand limitation of liability) set forth in the ServicesAgreement, or the scope of services, and terms and conditions providedto you ("Customer" or the "Company") in connection with the Agreement. This report providedin connection with the Servicesset forth in the Agreementshall be used by the Company only to the extent permittedunder the terms and conditions set forth in the Agreement.
This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes,nor may copiesbe delivered to any otherperson other than the Company,without FusionTech prior written consentin each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts FusionTech to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, businessmodel or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor shouldbe leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. FusionTech's position is that each companyand individual are responsible for their own due diligenceand continuous security.
FusionTech's goal is to help reduce the attack vectorsand the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by FusionTech is subject to dependencies and under continuing development. You agree that your access and/or use, includingbut 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 tokensare emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon,multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTSOR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS

3 | Global Overview

FusionTech uses certain vulnerability levels, these indicate how bad a certain issue is. The higher the risk, the more strictly it is recommended to correct the error before using the contract.

SEVERITY LEVEL	DESCRIPTION
CRITICAL	A vulnerability that can disrupt the contract functioning; creates a critical risk to the contract; required to be fixed immediately.
HIGH	A vulnerability that could affect the desired outcome of executing the contract with high impact; needs to be fixed with high priority.
MEDIUM	A vulnerability that could affect the desired outcome of executing the contract with medium impact in a specific scenario; needs to be fixed.
LOW	An issue that does not have a significant impact, can be considered as less important.

4 | Vulnerabilities Findings

The Smart Contract Weakness Classification Registry (SWC Registry) is an implementation of the weakness classication 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 specfic to smart contracts.

ID SWC-	Description	Status
100	Function Default Visibility	Passed
SWC-101	Integer Overflow and Underflow	Passed
SWC-102	Outdated Compiler Version	Passed
SWC-103	Floating Pragma	Passed
SWC-104	Unchecked Call Return Value	Passed
SWC-105	Unprotected Ether Withdrawal	Passed
SWC-106	Unprotected SELFDESTRUCT	Passed
SWC-107	Instruction Reentrancy	Passed
SWC-108	State Variable Default Visibility	Passed
SWC-109	Uninitialized Storage Pointer	Passed
SWC-110	Assert Violation	Passed
SWC-111	Use of Deprecated Solidity Functions	Passed
SWC-112	Delegatecall to Untrusted Callee	Passed
SWC-113	DoS with Failed Call	Passed
SWC-114	Transaction Order Dependence	Passed
SWC-115	Authorization through tx.origin	Passed

SWC-116	Block values as a proxy for time	Passed
SWC-117	Signature Malleability	Passed
SWC-118	Incorrect Constructor Name	Passed
SWC-119	Shadowing State Variables	Passed
SWC-120	Weak Sources of Randomness from Chain Attributes	Passed
SWC-121	Missing Protection against Signature Replay Attacks	Passed
SWC-122	Lack of Proper Signature Veri cation	Passed
SWC-123	Requirement Violation	Passed
SWC-124	Write to Arbitrary Storage Location	Passed
SWC-125	Incorrect Inheritance Order	Passed
SWC-126	Insuf cient Gas Grie ng	Passed
SWC-127	Arbitrary Jump with Function Type Variable	Passed
SWC-128	DoS With Block Gas Limit	Passed
SWC-129	Typographical Error	Passed
SWC-130	Right-To-Left-Override control character (U+202E)	Passed
SWC-131	Presence of unused variables	Passed
SWC-132	Unexpected Ether balance	Passed
SWC-133	Hash Collisions With Multiple Variable Length	Passed
SWC-134	Arguments Message call with hardcoded gas amount	Passed
SWC-135	Code With No Effects	Passed
SWC-136	Unencrypted Private Data On-Chain	Passed

Recommendation

Private keys belonging to the employer and/or contract owner should be stored properly. The initial asset allocation procedure should involve consultation with the community.

5 | Contract Privileges

5.1 | Maximum Fee Limit Check

FusionTech tests if the owner of the smart contract can set the transfer, buy or sell fee to 25% or more. It is bad practice to set the fees to 25% or more, because owners can prevent healthy trading or even stop trading when the fees are set too high.

ERROR	Description
Code	Centralization: Operator Fee
CEN-01	Manipulation

TYPE OF FEE	DESCRIPTION
Transacted without Tax	96%
Max buy fee	1%
Max sell fee	4%

5.2 | Contract Pausability Check

FusionTech tests if the owner of the smart contract has the ability to pause the contract. If this is the case, users can no longer interact with the smart contract; users can no longer trade the token.

Privilege Check	Description
Can owner pause the contract?	● Owner cannot pause the contract

5.3 | Max Transaction Amount Check

FusionTech tests if the owner of the smart contract can set the maximum amount of a transaction. If the transaction exceeds this limit, the transaction will revert. Owners could prevent normal transactions to take place if they abuse this function.

PRIVILEGE CHECK	DESCRIPTION
Can owner set max tx amount?	● Owner cannot set max transaction amount
Max buy fee	0%
Max sell fee	4%

5.4 | Exclude From Fees Check

FusionTech tests if the owner of the smart contract can exclude addresses from paying tax fees. If the owner of the smart contract can exclude from fees, they could set high tax fees and exclude themselves from fees and bene t from 0% trading fees. However, some smart contracts require this function to exclude routers, dex, cex or other contracts / wallets from fees.

Privilege Check	Description
Can owner exclude from fees?	● Owner can exclude from fees

FUNCTION

```
function excludeFromFees(address account, bool excluded) external onlyOwner{
_isExcludedFromFees[account] = excluded;

 emit ExcludeFromFees(account, excluded); }
```

5.5 | Ability to Mint Check

FusionTech tests if the owner of the smart contract can mint new tokens. If the contract contains a mint function, we refer to the token's total supply as non- xed, allowing the token owner to "mint" more tokens whenever they want.

A mint function in the smart contract allows minting tokens at a later stage. A method to disable minting can also be added to stop the minting process irreversibly.

Minting tokens is done by sending a transaction that creates new tokens inside of the token smart contract. With the help of the smart contract function, an unlimited number of tokens can be created without spending additional energy or money.

Privilege Check	Description
Can owner mint?	● Owner cannot mint new tokens

5.6 | Ability to Blacklist Check

FusionTech tests if the owner of the smart contract can blacklist accounts from interacting with the smart contract. Blacklisting methods allow the contract owner to enter wallet addresses which are not allowed to interact with the smart contract.

This method can be abused by token owners to prevent certain / all holders from trading the token. However, blacklists might be good for tokens that want to rule out certain addresses from interacting with a smart contract.

Privilege Check	Description
Can owner blacklist?	● Owner cannot blacklist an wallet/address

6 | Contract Snapshot

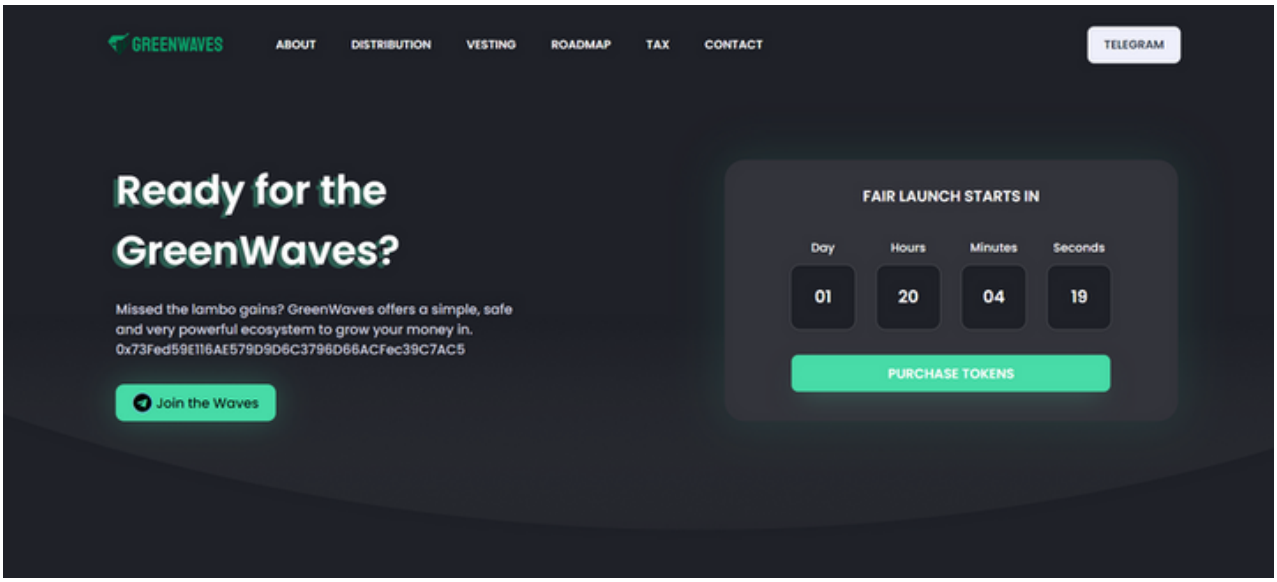
This is how the constructor of the contract looked at the time of auditing the smart contract.

```
contract DxFeeToken is Context, IERC20, Ownable {
 using SafeMath for uint256;
          address          private          dead          =
0x000000000000000000000000000000000000dEaD;
 uint256 public maxLiqFee = 10;
 uint256 public maxTaxFee = 10;
 uint256 public maxDevFee = 10;
 uint256 public minMxTxPercentage = 50;
 uint256 public maxSellTaxFee = 20;
 uint256 public maxSellLiqFee = 20;

 mapping (address => uint256) private _rOwned;
 mapping (address => uint256) private _tOwned;
 mapping (address => mapping (address => uint256)) private
_allowances;
 mapping (address => bool) private _isExcludedFromFee;
 mapping (address => bool) private _isExcluded;
 mapping (address => bool) private _isdevWallet;
```

7 | Website Review

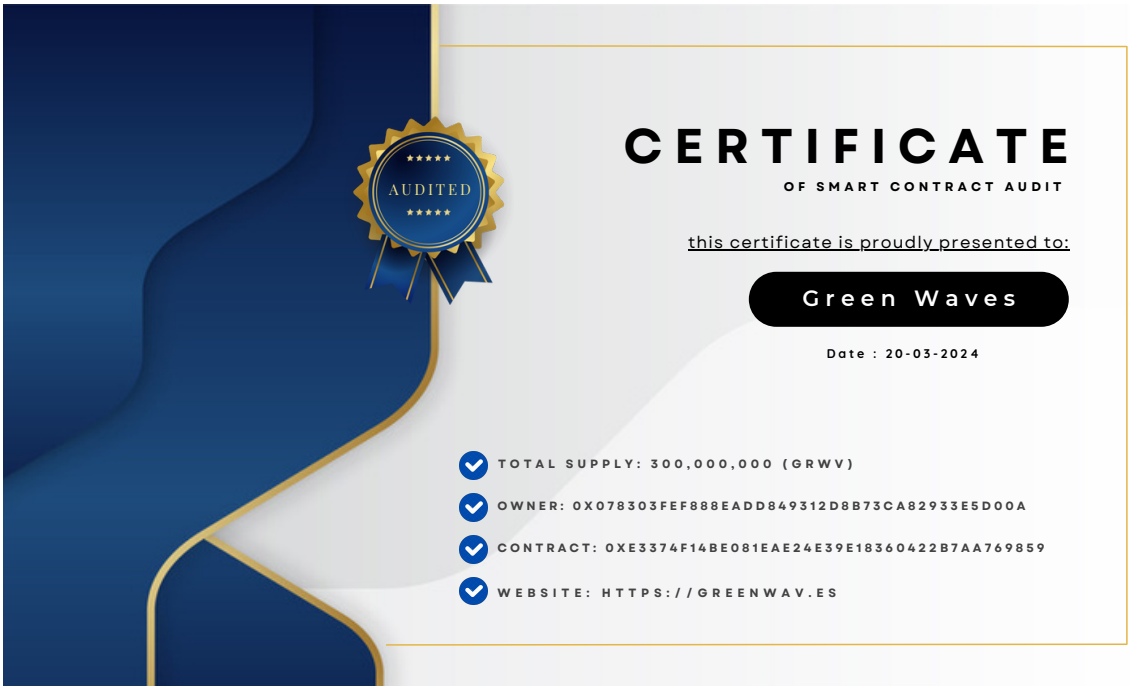
FusionTech checks the website completely manually and looks for visual, technical and textual errors. We also look at the security, speed and accessibility of the website. In short, a complete check to see if the website meets the current standard of the web development industry.



Type of check	Description
Mobile friendly?	The website is mobile friendly
Contains jQuery errors?	The website does not contain jQuery errors
Is SSL secured?	The website is SSL secured
Contains spelling errors?	The website does not contain spelling errors

8 | Certifcate of Proof

- Smart Contract Audited
- KYC Verified



Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanismto relocate funds.

Logical Issue

Logical Issue findingsdetail a fault in the logic of the linked code, such as an incorrect notion on how block.times tamp works.

Volatile Code

Volatile Code findingsrefer to segmentsof code that behave unexpectedly on certain edge cases that may resultin a vulnerability.

Coding Style

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

Inconsistency

Inconsistency findings referto 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.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexa-decimal encoded and is the same as the output of the Linux "sha256sum" commandagainst the target file.

About

Founded in 2022 by leading academics in the field of Computer Science, FusionTech is going to be a leading blockchain security company that serves to verify the security and correctness of smart contracts KYC and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of ourclients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.

FusionTech

https://fusiontech.network

https://twitter.com/fusiontechh

https://t.me/fusiontechchat

https://github.com/fusiontechofficial