



**SOLIDProof**  
*Bring trust into your projects*

**Blockchain Security | Smart Contract Audits | KYC**

MADE IN GERMANY

**Meetway**

**Audit**

**Security Assessment**

**06. May, 2022**

**For**



Disclaimer	3
Description	5
Project Engagement	5
Logo	5
Contract Link	5
Methodology	8
Used Code from other Frameworks/Smart Contracts (direct imports)	9
Tested Contract Files	10
Source Lines	12
Risk Level	12
Capabilities	13
Inheritance Graph	14
CallGraph	15
Scope of Work/Verify Claims	16
Modifiers and public functions	22
Source Units in Scope	25
Critical issues	27
High issues	27
Medium issues	27
Low issues	27
Informational issues	28
Audit Comments	28
SWC Attacks	29

# Disclaimer

SolidProof.io reports are not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team. SolidProof.io do not cover testing or auditing the integration with external contract or services (such as Uniswap, Uniswap, PancakeSwap etc’...)

**SolidProof.io Audits do not provide any warranty or guarantee regarding the absolute bug- free nature of the technology analyzed, nor do they provide any indication of the technology proprietors. SolidProof Audits should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.**

SolidProof.io Reports represent an extensive auditing 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. SolidProof’s position is that each company and individual are responsible for their own due diligence and continuous security. SolidProof in no way claims any guarantee of security or functionality of the technology we agree to analyze.

Version	Date	Description
1.0	28. April 2022	<ul style="list-style-type: none"><li>• Layout project</li><li>• Automated- /Manual-Security Testing</li><li>• Summary</li></ul>
1.1	11. May 2022	<ul style="list-style-type: none"><li>• Reaudit</li></ul>

## **Network**

Binance Smart Chain (BEP20)

## **Website**

<https://meetway.io/>

## **Telegram**

<https://t.me/MEETWAYofficial>

## **Twitter**

<https://twitter.com/meetwayofficial>

## **Discord**

<https://discord.gg/meetway>

## **TikTok**

<https://www.tiktok.com/@meetwayofficial>

## Description

MeetWay is a mobile M2E app that allows people to connect, exercise, and earn.

**MeetWay's** ecosystem is a combination of music and the real world. This encourages users to engage more with the people and community around them as well as pushes them to build a healthy lifestyle.

Accordingly, MeetWay introduces the first Metaverse e-fitness, complete with unique features that haven't seen in any other health app. Running/cycling with teammates, weather notifications, terrain, achievement systems, and other features of the app will show its utility while also giving entertainment, encouraging users to practice every day. MeetWay is special in that it allows us to adopt healthy habits while still earning money.

## Project Engagement

During the 25th of April 2022, **Meetway Team** engaged Solidproof.io to audit smart contracts that they created. The engagement was technical in nature and focused on identifying security flaws in the design and implementation of the contracts. They provided Solidproof.io with access to their code repository and whitepaper.

## Logo



## Contract Link v1.0

- Github
  - <https://github.com/meetway2022/meetway>
  - Commit: fcef61cc26b69f084d7a76e64b40a52884802f8b
- Bsc
  - <https://bscscan.com/address/0x1bB7a6024a4c4878cA870B58cC64c92b98b945Fc>

## v1.1

- Github
  - <https://github.com/meetway2022/meetway>
  - Commit: 54985435dba90227dce2679fb7a12db25b0b89ec



# Vulnerability & Risk Level

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	Value	Vulnerability	Risk (Required Action)
<b>Critical</b>	9 - 10	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.
<b>High</b>	7 - 8.9	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.
<b>Medium</b>	4 - 6.9	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.	Implementation of corrective actions in a certain period.
<b>Low</b>	2 - 3.9	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.
<b>Informational</b>	0 - 1.9	A vulnerability that have informational character but is not effecting any of the code.	An observation that does not determine a level of risk

# 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

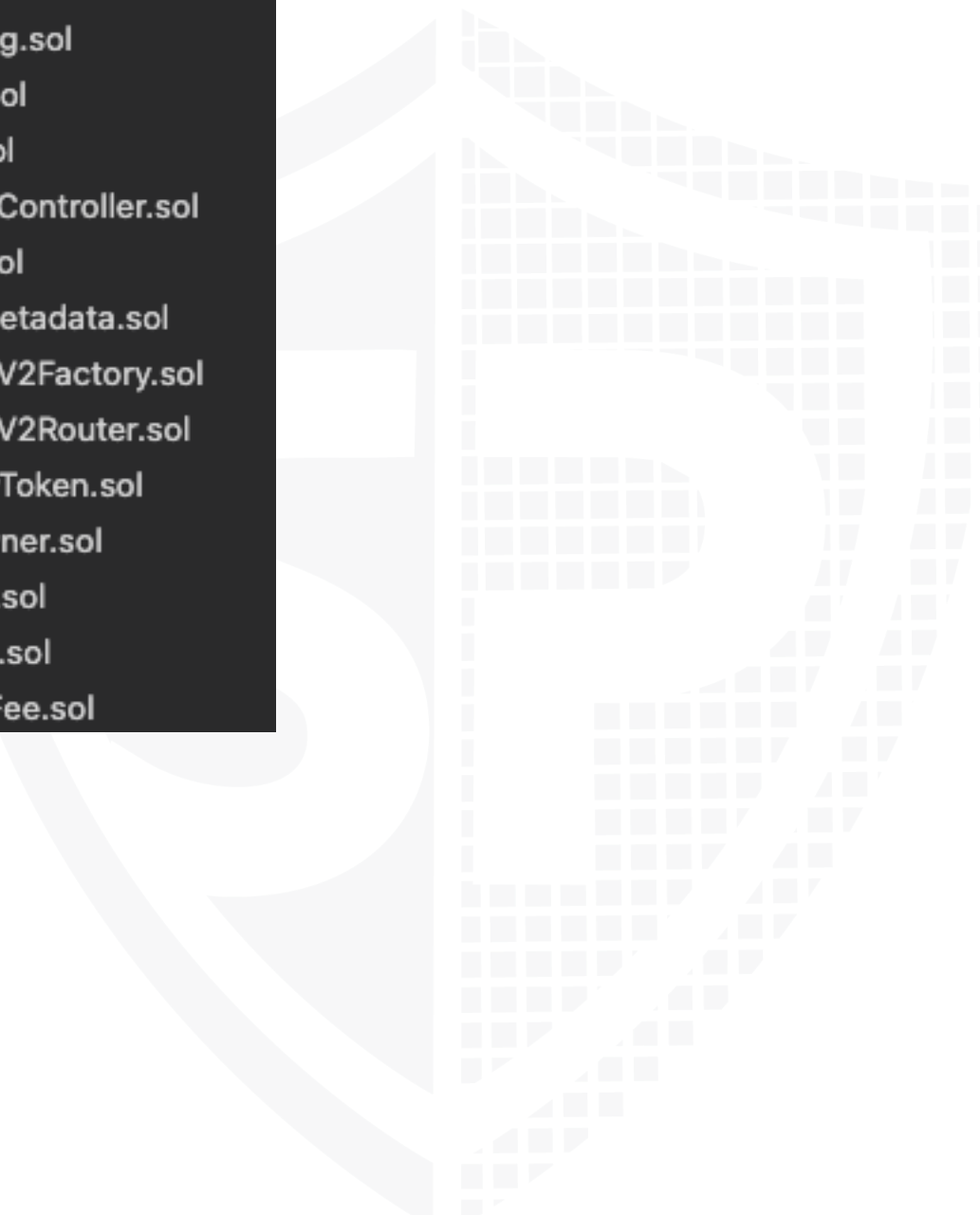
The auditing process follows a routine series of steps:

1. Code review that includes the following:
  - i) Review of the specifications, sources, and instructions provided to SolidProof to make sure we understand the size, scope, and functionality of the smart contract.
  - ii) Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
  - iii) Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to SolidProof describe.
2. Testing and automated analysis that includes the following:
  - i) 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.
  - ii) 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.



## Used Code from other Frameworks/Smart Contracts (direct imports)

Imported packages:



```
Context.sol
DexListing.sol
DexPair.sol
ERC20.sol
GasPriceController.sol
IERC20.sol
IERC20Metadata.sol
IUniswapV2Factory.sol
IUniswapV2Router.sol
MeetWayToken.sol
OriginOwner.sol
Ownable.sol
Pausable.sol
TransferFee.sol
```

## Tested Contract Files

This audit covered the following files listed below with a SHA-1 Hash.

*A file with a different Hash has been modified, intentionally or otherwise, after the security review. A different Hash could be (but not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of this review.*

### v1.0

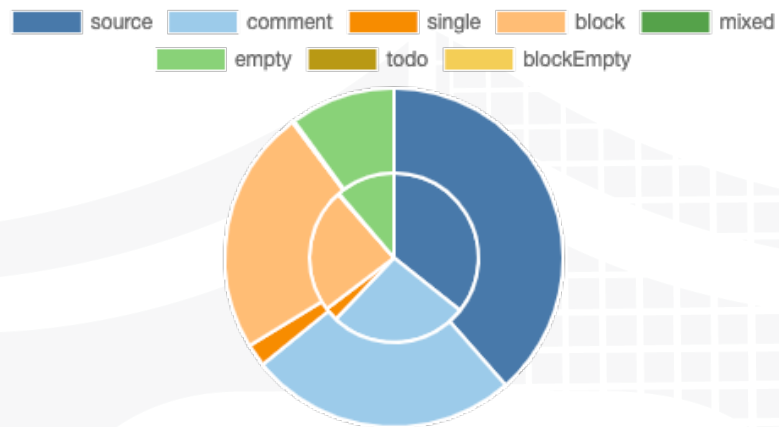
File Name	SHA-1 Hash
contracts/Context.sol	164272374e4a7a8499d9f75af6b33f37a75b61e8
contracts/IUniswapV2Factory.sol	fec5564db2888e0a6c0f495000ff6cbc7be2c732
contracts/DexListing.sol	89adc16ef4b1a9700d1ac776994131ce5ca2e7a3
contracts/IERC20Metadata.sol	d42a8bd1ccbe0bcf085972b58f6095da433370b4
contracts/TransferFee.sol	b3263360bc53b3d7d3cd4c984e83b75c151f3366
contracts/OriginOwner.sol	6a71a7567ae1d672a0e445113ca0e15c96322913
contracts/DexPair.sol	fc3e1cc9b9d44a59e4cd7291dd78766ecb922b6e
contracts/GasPriceController.sol	4bfa65c8d94be6b1849b724fd9100b41134d0709
contracts/IUniswapV2Router.sol	65717068028153627c9c07e27e84b8f4061e8172
contracts/Ownable.sol	7db188fad5e4ea3e0b1a6741c83f8380653d2801
contracts/Pausable.sol	d42ae199d2cfb083f59069b5de92fd36ae28e7c5
contracts/MeetWayToken.sol	a6bc5a8e2a896b0bb63265099cf9786cdec76e
contracts/ERC20.sol	0356a839afdce94958f278a28feb56cfe35fe173
contracts/IERC20.sol	6c31117e13085c9011db480856e8d7a99998000d

## v1.1

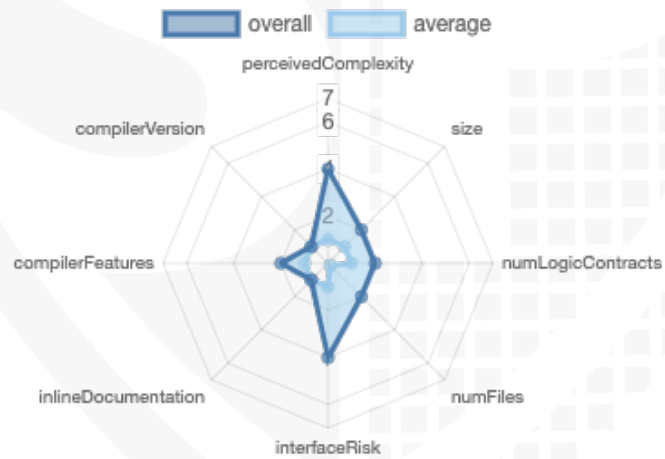
File Name	SHA-1 Hash
contracts/Context.sol	164272374e4a7a8499d9f75af6b33f37a75b61e8
contracts/IUniswapV2Factory.sol	fec5564db2888e0a6c0f495000ff6cbc7be2c732
contracts/DexListing.sol	89adc16ef4b1a9700d1ac776994131ce5ca2e7a3
contracts/IERC20Metadata.sol	d42a8bd1ccbe0bcf085972b58f6095da433370b4
contracts/TransferFee.sol	3cfe3f8c933e53eac2b6609ff688cfc93699662a
contracts/DexPair.sol	fc3e1cc9b9d44a59e4cd7291dd78766ecb922b6e
contracts/GasPriceController.sol	4bfa65c8d94be6b1849b724fd9100b41134d0709
contracts/IUniswapV2Router.sol	65717068028153627c9c07e27e84b8f4061e8172
contracts/Ownable.sol	7db188fad5e4ea3e0b1a6741c83f8380653d2801
contracts/MeetWayToken.sol	0c503434dd3deffe5b08c37f3be0c924bd8f455
contracts/ERC20.sol	0356a839afdce94958f278a28feb56cfe35fe173
contracts/IERC20.sol	6c31117e13085c9011db480856e8d7a99998000d

# Metrics

## Source Lines v1.0



## Risk Level v1.0



## Capabilities

### Components

Version	Contracts	Libraries	Interfaces	Abstract
1.0	6	1	5	3

### Exposed Functions

*This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.*

Version	Public	Payable
1.0	46	0

Version	External	Internal	Private	Pure	View
1.0	22	75	3	1	25

### State Variables

Version	Total	Public
1.0	25	5

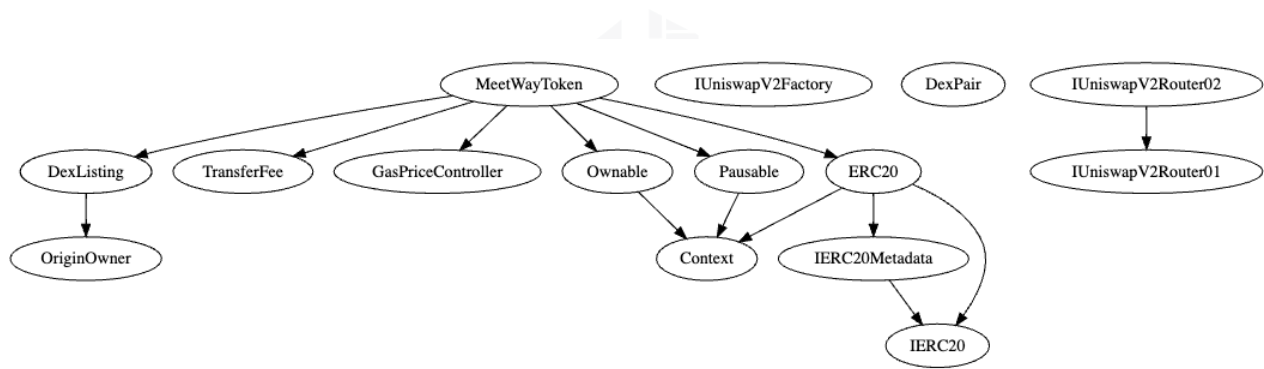
## Capabilities

Version	Solidity Versions observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
1.0	<code>^0.8.4</code> <code>^0.8.0</code>			yes (1 asm blocks)	

Version	Transfers ETH	Low-Level Calls	DelegateCall	Uses Hash Functions	EC Recover	New/Create/Create2
---------	---------------	-----------------	--------------	---------------------	------------	--------------------

1.0	yes			yes		
-----	-----	--	--	-----	--	--

## Inheritance Graph v1.0





## Scope of Work/Verify Claims

The above token Team provided us with the files that needs to be tested (Github, Bscscan, Etherscan, files, etc.). The scope of the audit is the main contract (usual the same name as team appended with .sol).

We will verify the following claims:

1. Correct implementation of Token standard
2. Deployer cannot mint any new tokens
3. Deployer cannot burn or lock user funds
4. Deployer cannot pause the contract
5. Overall checkup (Smart Contract Security)

### Correct implementation of Token standard

ERC20				
Function	Description	Exist	Tested	Verified
TotalSupply	Provides information about the total token supply	✓	✓	✓
BalanceOf	Provides account balance of the owner's account	✓	✓	✓
Transfer	Executes transfers of a specified number of tokens to a specified address	✓	✓	✓
TransferFrom	Executes transfers of a specified number of tokens from a specified address	✓	✓	✓
Approve	Allow a spender to withdraw a set number of tokens from a specified account	✓	✓	✓
Allowance	Returns a set number of tokens from a spender to the owner	✓	✓	✓



## Write functions of contract v1.0

addBlackList  
addBlackLists  
removeBlackList  
removeBlackLists  
setMaxAmount  
setMaxGasPrice  
setTransferFee  
pause  
unpause  
withdrawBalance  
withdrawTokens

renounceOwnership  
transferOwnership

transfer  
approve  
transferFrom  
increaseAllowance  
decreaseAllowance

## Deployer cannot mint any new tokens

Name	Exist	Tested	Status
Deployer cannot mint	✓	✓	✓
Max / Total Supply	200000		



## Deployer cannot burn or lock user funds

Name	Exist	Tested	Status
Deployer cannot lock	✓	✓	✗
Deployer cannot burn	✓	✓	✓

Comments:

**v1.0**

- Owner can lock user funds by
  - blacklisting addresses

## Deployer cannot pause the contract

Name	Exist	Tested	Status
Deployer cannot pause	—	—	—

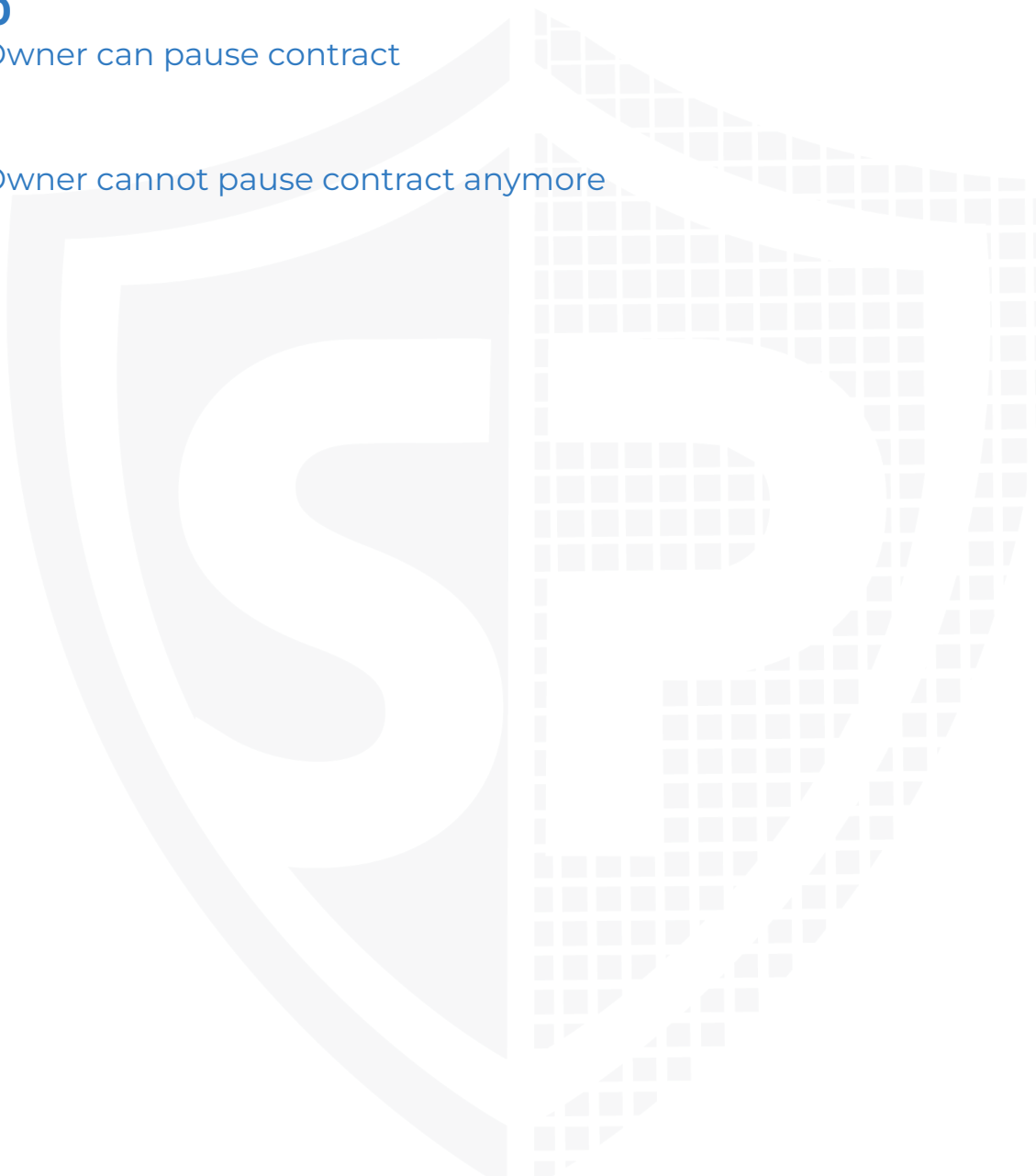
Comments:

### v1.0

- Owner can pause contract

### v1.1

- Owner cannot pause contract anymore



## Overall checkup (Smart Contract Security)



























Tested	Verified
✓	✓

### Legend

Attribute	Symbol
Verified / Checked	✓
Partly Verified	⚠
Unverified / Not checked	✗
Not available	—

# Modifiers and public functions

v1.0













▼  <b>addBlackList</b>  onlyOwner	▼  <b>renounceOwnership</b>  onlyOwner
▼  <b>addBlackLists</b>  onlyOwner	▼  <b>transferOwnership</b>  onlyOwner
▼  <b>removeBlackList</b>  onlyOwner	
▼  <b>removeBlackLists</b>  onlyOwner	
▼  <b>setMaxAmount</b>  onlyOwner	
▼  <b>setMaxGasPrice</b>  onlyOwner	
▼  <b>setTransferFee</b>  onlyOwner	
▼  <b>pause</b>  onlyOwner	
▼  <b>unpause</b>  onlyOwner	
▼  <b>withdrawBalance</b>  onlyOwner	
▼  <b>withdrawTokens</b>  onlyOwner	

## Comments

- Deployer can set following state variables without any limitations
  - `_transferFee.to`
  - `_transferFee.buy`
  - `_transferFee.sell`
  - `_transferFee.normal`
- Deployer can enable/disable following state variables
  - `blackListedList`
  - `_paused`

- Existing Modifiers
  - onlyValidGasPrice
  - onlyOriginOwner
  - onlyOwner
  - whenNotPaused
  - whenPaused
- Max tx amount has no functionality

## v1.1

<ul style="list-style-type: none"> <li>✓  <b>addBlackList</b> <ul style="list-style-type: none"> <li>Ⓜ onlyOwner</li> </ul> </li> <li>✓  <b>addBlackLists</b> <ul style="list-style-type: none"> <li>Ⓜ onlyOwner</li> </ul> </li> <li>✓  <b>removeBlackList</b> <ul style="list-style-type: none"> <li>Ⓜ onlyOwner</li> </ul> </li> <li>✓  <b>removeBlackLists</b> <ul style="list-style-type: none"> <li>Ⓜ onlyOwner</li> </ul> </li> <li>✓  <b>addExcludeFee</b> <ul style="list-style-type: none"> <li>Ⓜ onlyOwner</li> </ul> </li> <li>✓  <b>removeExcludeFee</b> <ul style="list-style-type: none"> <li>Ⓜ onlyOwner</li> </ul> </li> <li>✓  <b>setMaxGasPrice</b> <ul style="list-style-type: none"> <li>Ⓜ onlyOwner</li> </ul> </li> <li>✓  <b>setTransferFee</b> <ul style="list-style-type: none"> <li>Ⓜ onlyOwner</li> </ul> </li> <li>✓  <b>withdrawBalance</b> <ul style="list-style-type: none"> <li>Ⓜ onlyOwner</li> </ul> </li> <li>✓  <b>withdrawTokens</b> <ul style="list-style-type: none"> <li>Ⓜ onlyOwner</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>✓  <b>renounceOwnership</b> <ul style="list-style-type: none"> <li>Ⓜ onlyOwner</li> </ul> </li> <li>✓  <b>transferOwnership</b> <ul style="list-style-type: none"> <li>Ⓜ onlyOwner</li> </ul> </li> </ul>
--	--

## Comments

- Deployer can enable/disable following state variables
  - blackListedList
- Existing Modifiers
  - onlyValidGasPrice

- `onlyOwner`























**Please check if an `OnlyOwner` or similar restrictive modifier has been forgotten.**

























# Source Units in Scope

## v1.0

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	contracts/Context.sol	1	————	24	24	9	12	1	————
	contracts/IUniswapV2Factory.sol	————	1	9	6	3	1	3	————
	contracts/DexListing.sol	1	————	118	92	65	12	32	————
	contracts/IERC20Metadata.sol	————	1	28	17	4	16	9	
	contracts/TransferFee.sol	1	————	77	55	43	1	20	
	contracts/OriginOwner.sol	1	————	73	51	35	5	18	————
	contracts/DexPair.sol	1	————	42	31	17	5	21	
	contracts/GasPriceController.sol	1	————	37	29	21	1	8	————
	contracts/IUniswapV2Router.sol	————	2	10	7	4	1	6	————
	contracts/Ownable.sol	1	————	76	76	28	38	23	————
	contracts/Pausable.sol	1	————	91	91	29	51	16	————
	contracts/MeetWayToken.sol	1	————	168	144	103	17	104	
	contracts/ERC20.sol	1	————	356	336	103	194	80	
	contracts/IERC20.sol	————	1	82	27	17	58	13	
	<b>Totals</b>	<b>10</b>	<b>5</b>	<b>1191</b>	<b>986</b>	<b>481</b>	<b>412</b>	<b>354</b>	

## v1.1

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	contracts/Context.sol	1	————	24	24	9	12	1	————
	contracts/IUniswapV2Factory.sol	————	1	9	6	3	1	3	————
	contracts/DexListing.sol	1	————	118	92	65	12	32	————
	contracts/IERC20Metadata.sol	————	1	28	17	4	16	9	
	contracts/TransferFee.sol	1	————	77	55	43	1	20	
	contracts/DexPair.sol	1	————	42	31	17	5	21	
	contracts/GasPriceController.sol	1	————	37	29	21	1	8	————
	contracts/IUniswapV2Router.sol	————	2	10	7	4	1	6	————
	contracts/Ownable.sol	1	————	76	76	28	38	23	————
	contracts/MeetWayToken.sol	1	————	164	140	112	11	107	
	contracts/ERC20.sol	1	————	356	336	103	194	80	
	contracts/IERC20.sol	————	1	82	27	17	58	13	
	<b>Totals</b>	<b>8</b>	<b>5</b>	<b>1023</b>	<b>840</b>	<b>426</b>	<b>350</b>	<b>323</b>	

## Legend

Attribute	Description
Lines	total lines of the source unit

nLines	normalized lines of the source unit (e.g. normalizes functions spanning multiple lines)
nSLOC	normalized source lines of code (only source-code lines; no comments, no blank lines)
Comment Lines	lines containing single or block comments
Complexity Score	a custom complexity score derived from code statements that are known to introduce code complexity (branches, loops, calls, external interfaces, ...)



# Audit Results

## AUDIT PASSED

### Critical issues

No critical issues

### High issues

No high issues

### Medium issues

No medium issues

### Low issues

Issue	File	Type	Line	Description
#1	Main	Contract doesn't import npm packages from source (like OpenZeppelin etc.)	-	We recommend to import all packages from npm directly without flatten the contract. Functions could be modified or can be susceptible to vulnerabilities
#2	All	A floating pragma is set	At the top of source file	The current pragma Solidity directive is „^0.8.4”.
#3	MeetWaiToken	Local variables shadowing	59, 138	Rename the local variables that shadow another component
#4	All	Remove imported packages	Where it was imported	Import not existing imports. For example:  OriginOwner.sol  File is not existing anymore and remove everything what is related to the contract

## Informational issues

Issue	File	Type	Line	Description
#1	Transfer Fee	Unused state variables	9	Remove unused state variables
#2	Main	NatSpec documentation missing	-	If you started to comment your code, also comment all other functions, variables etc.

## Audit Comments

We recommend you to use the special form of comments (NatSpec Format, Follow link for more information <https://docs.soliditylang.org/en/v0.5.10/natspec-format.html>) for your contracts to provide rich documentation for functions, return variables and more. This helps investors to make clear what that variables, functions etc. do.

### 11. May 2022:

- Read whole report for more information

## SWC Attacks

ID	Title	Relationships	Status
<a href="#">SW C-1 36</a>	Unencrypted Private Data On-Chain	<a href="#">CWE-767: Access to Critical Private Variable via Public Method</a>	PASSED
<a href="#">SW C-1 35</a>	Code With No Effects	<a href="#">CWE-1164: Irrelevant Code</a>	PASSED
<a href="#">SW C-1 34</a>	Message call with hardcoded gas amount	<a href="#">CWE-655: Improper Initialization</a>	PASSED
<a href="#">SW C-1 33</a>	Hash Collisions With Multiple Variable Length Arguments	<a href="#">CWE-294: Authentication Bypass by Capture-replay</a>	PASSED
<a href="#">SW C-1 32</a>	Unexpected Ether balance	<a href="#">CWE-667: Improper Locking</a>	PASSED
<a href="#">SW C-1 31</a>	Presence of unused variables	<a href="#">CWE-1164: Irrelevant Code</a>	NOT PASSED
<a href="#">SW C-1 30</a>	Right-To-Left-Override control character (U+202E)	<a href="#">CWE-451: User Interface (UI) Misrepresentation of Critical Information</a>	PASSED
<a href="#">SW C-1 29</a>	Typographical Error	<a href="#">CWE-480: Use of Incorrect Operator</a>	PASSED
<a href="#">SW C-1 28</a>	DoS With Block Gas Limit	<a href="#">CWE-400: Uncontrolled Resource Consumption</a>	PASSED

<a href="#">SW C-1 27</a>	Arbitrary Jump with Function Type Variable	<a href="#">CWE-695: Use of Low-Level Functionality</a>	PASSED
<a href="#">SW C-1 25</a>	Incorrect Inheritance Order	<a href="#">CWE-696: Incorrect Behavior Order</a>	PASSED
<a href="#">SW C-1 24</a>	Write to Arbitrary Storage Location	<a href="#">CWE-123: Write-what-where Condition</a>	PASSED
<a href="#">SW C-1 23</a>	Requirement Violation	<a href="#">CWE-573: Improper Following of Specification by Caller</a>	PASSED
<a href="#">SW C-1 22</a>	Lack of Proper Signature Verification	<a href="#">CWE-345: Insufficient Verification of Data Authenticity</a>	PASSED
<a href="#">SW C-1 21</a>	Missing Protection against Signature Replay Attacks	<a href="#">CWE-347: Improper Verification of Cryptographic Signature</a>	PASSED
<a href="#">SW C-1 20</a>	Weak Sources of Randomness from Chain Attributes	<a href="#">CWE-330: Use of Insufficiently Random Values</a>	PASSED
<a href="#">SW C-11 9</a>	Shadowing State Variables	<a href="#">CWE-710: Improper Adherence to Coding Standards</a>	NOT PASSED
<a href="#">SW C-11 8</a>	Incorrect Constructor Name	<a href="#">CWE-665: Improper Initialization</a>	PASSED
<a href="#">SW C-11 7</a>	Signature Malleability	<a href="#">CWE-347: Improper Verification of Cryptographic Signature</a>	PASSED

<a href="#">SW C-11 6</a>	Timestamp Dependence	<a href="#">CWE-829: Inclusion of Functionality from Untrusted Control Sphere</a>	<b>PASSED</b>
<a href="#">SW C-11 5</a>	Authorization through tx.origin	<a href="#">CWE-477: Use of Obsolete Function</a>	<b>PASSED</b>
<a href="#">SW C-11 4</a>	Transaction Order Dependence	<a href="#">CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')</a>	<b>PASSED</b>
<a href="#">SW C-11 3</a>	DoS with Failed Call	<a href="#">CWE-703: Improper Check or Handling of Exceptional Conditions</a>	<b>PASSED</b>
<a href="#">SW C-11 2</a>	Delegatecall to Untrusted Callee	<a href="#">CWE-829: Inclusion of Functionality from Untrusted Control Sphere</a>	<b>PASSED</b>
<a href="#">SW C-11 1</a>	Use of Deprecated Solidity Functions	<a href="#">CWE-477: Use of Obsolete Function</a>	<b>PASSED</b>
<a href="#">SW C-11 0</a>	Assert Violation	<a href="#">CWE-670: Always-Incorrect Control Flow Implementation</a>	<b>PASSED</b>
<a href="#">SW C-1 09</a>	Uninitialized Storage Pointer	<a href="#">CWE-824: Access of Uninitialized Pointer</a>	<b>PASSED</b>
<a href="#">SW C-1 08</a>	State Variable Default Visibility	<a href="#">CWE-710: Improper Adherence to Coding Standards</a>	<b>PASSED</b>
<a href="#">SW C-1 07</a>	Reentrancy	<a href="#">CWE-841: Improper Enforcement of Behavioral Workflow</a>	<b>PASSED</b>
<a href="#">SW C-1 06</a>	Unprotected SELFDESTRUCT Instruction	<a href="#">CWE-284: Improper Access Control</a>	<b>PASSED</b>

<a href="#">SW</a> <a href="#">C-1</a> <a href="#">05</a>	Unprotected Ether Withdrawal	<a href="#">CWE-284: Improper Access Control</a>	<b>PASSED</b>
<a href="#">SW</a> <a href="#">C-1</a> <a href="#">04</a>	Unchecked Call Return Value	<a href="#">CWE-252: Unchecked Return Value</a>	<b>PASSED</b>
<a href="#">SW</a> <a href="#">C-1</a> <a href="#">03</a>	Floating Pragma	<a href="#">CWE-664: Improper Control of a Resource Through its Lifetime</a>	<b>NOT PASSED</b>
<a href="#">SW</a> <a href="#">C-1</a> <a href="#">02</a>	Outdated Compiler Version	<a href="#">CWE-937: Using Components with Known Vulnerabilities</a>	<b>PASSED</b>
<a href="#">SW</a> <a href="#">C-1</a> <a href="#">01</a>	Integer Overflow and Underflow	<a href="#">CWE-682: Incorrect Calculation</a>	<b>PASSED</b>
<a href="#">SW</a> <a href="#">C-1</a> <a href="#">00</a>	Function Default Visibility	<a href="#">CWE-710: Improper Adherence to Coding Standards</a>	<b>PASSED</b>



The logo features the word "SolidProof" in a white, elegant script font. The "P" is particularly large and stylized, with a long horizontal stroke that extends to the left. The background is a solid blue color with a faint, large shield emblem. The shield has a grid-like pattern on its right side and a solid blue area on its left side.

SolidProof

**Blockchain Security | Smart Contract Audits | KYC**

A small horizontal bar representing the German flag, with black, red, and gold stripes.

MADE IN GERMANY