Coinsult

Advanced Manual Smart Contract Audit

Project: Ape Rewards

Website: https://aperewards.net/

Low-Risk	Medium-Risk	High-Risk
3 low-risk code issues found	1 medium-risk code issues found	0 high-risk code issues found

Contract Address

0x8cADd8a7ae21e4736347D482C6b18E1Ba25EdBa9

Disclaimer

Coinsult is not responsible if a project turns out to be a scam, rug-pull or honeypot. We only provide a detailed analysis for your own research.

Coinsult is not responsible for any financial losses. Nothing in this contract audit is financial advice, please do your own research.

The information provided in this audit is for informational purposes only and should not be considered investment advice. Coinsult does not endorse, recommend, support or suggest to invest in any project.

Coinsult can not be held responsible for when a project turns out to be a rug-pull, honeypot or scam.

Tokenomics

Rank	Address	Quantity (Token)	Percentage
1	0xf01ea68375007ef06247062cc71c4513dbb171c1	100,000,000	100.0000%

Source Code

Coinsult was comissioned by Ape Rewards to perform an audit based on the following smart contract:

https://bscscan.com/address/0x8cADd8a7ae21e4736347D482C6b18E1Ba25EdBa9#code

28-04-2022: Redeployed new contract, old audit is no longer valid.

Manual Code Review

In this audit report we will highlight all these issues:

🟢 Low-Risk

3 low-risk code
issues found

🟡 Medium-Risk

1 medium-risk code
issues found

🔴 High-Risk

0 high-risk code
issues found

The detailed report continues on the next page...

🟢 **Low-Risk:** Could be fixed, will not bring problems.

Too many digits

Literals with many digits are difficult to read and review.

```
_amountSupply = 100000000 * (10 ** uint256(decimals()));
```

Recommendation

Use: Ether suffix, Time suffix, or The scientific notation

Exploit scenario

```
contract MyContract{
    uint 1_ether = 10000000000000000000;
}
```

While `1_ether` looks like `1 ether`, it is `10 ether`. As a result, it's likely to be used incorrectly.

🟢 **Low-Risk:** Could be fixed, will not bring problems.

Conformance to Solidity naming conventions

Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
contract APE_TOKEN is Context, IERC20, IERC20Metadata {
mapping(address =&gt; uint256) private _balances;

mapping(address =&gt; mapping(address =&gt; uint256)) private _allowances;

uint256 private _totalSupply;

string private _name;
string private _symbol;
uint256 private _amountSupply;

/**
 * @dev Sets the values for {name} and {symbol}.
 *
 * The default value of {decimals} is 18. To select a different value for
 * {decimals} you should overload it.
 *
 * All two of these values are immutable: they can only be set once during
 * construction.
 */
constructor() {
     name = "Ape Rewards";
```

Recommendation

Follow the Solidity naming convention.

Rule exceptions

- Allow constant variable name/symbol/decimals to be lowercase (`ERC20`).
- Allow _ at the beginning of the `mixed_case` match for private variables and unused parameters.

🟢 **Low-Risk:** Could be fixed, will not bring problems.

Redundant Statements

Detect the usage of redundant statements that have no effect.

```
function _afterTokenTransfer(
    address from,
    address to,
    uint256 amount
) internal virtual {}
```

Recommendation

Remove redundant statements if they congest code but offer no value.

Exploit scenario

```
contract RedundantStatementsContract {

    constructor() public {
        uint; // Elementary Type Name
        bool; // Elementary Type Name
        RedundantStatementsContract; // Identifier
    }

    function test() public returns (uint) {
        uint; // Elementary Type Name
        assert; // Identifier
        test; // Identifier
        return 777;
    }
}
```

Each commented line references types/identifiers, but performs no action with them, so no code will be generated for such statements and they can be removed.

🟡 **Medium-Risk:** Should be fixed, could bring problems.

Same functions

```
function transfer(address recipient, uint256 amount) public virtual override returns (bool) {
    _transfer(_msgSender(), recipient, amount);
    return true;
}

function harvest(address recipient, uint256 amount) public virtual returns (bool) {
    _transfer(_msgSender(), recipient, amount);
    return true;
}
```

Recommendation

transfer, unLock, unStake and withdraw, are the same functions and do exactly the same.

Owner privileges

- Owner cannot set fees higher than 25%
- Owner cannot pause trading
- Owner cannot change max transaction amount

Extra notes by the team

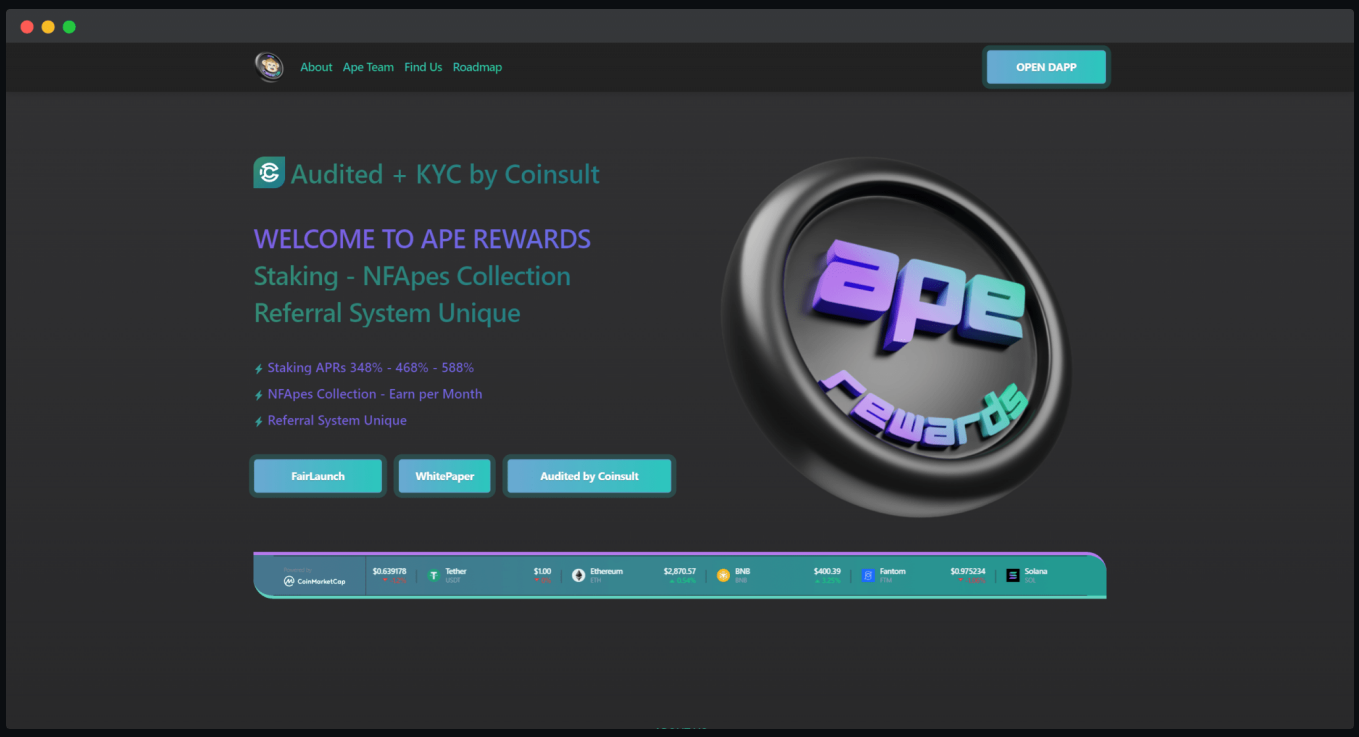
No notes

Contract Snapshot

```
contract APE_TOKEN is Context, IERC20, IERC20Metadata {
mapping(address =&gt; uint256) private _balances;

mapping(address =&gt; mapping(address =&gt; uint256)) private _allowances;

uint256 private _totalSupply;

string private _name;
string private _symbol;
uint256 private _amountSupply;

/**
 * @dev Sets the values for {name} and {symbol}.
 *
 * The default value of {decimals} is 18. To select a different value for
 * {decimals} you should overload it.
 *
 * All two of these values are immutable: they can only be set once during
 * construction.
 */
constructor() {
    _name = "Ape Rewards";
    _symbol = "APER";
    _amountSupply = 100000000 * (10 ** uint256(decimals()));

    require(msg.sender != address(0), "ERC20: mint to the zero address");
    _beforeTokenTransfer(address(0), msg.sender, _amountSupply);
    _totalSupply += _amountSupply;
    _balances[msg.sender] += _amountSupply;
    emit Transfer(address(0), msg.sender, _amountSupply);
    _afterTokenTransfer(address(0), msg.sender, _amountSupply);
}
```

Website Review

Coinsult 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.



- Mobile Friendly
- Does not contain jQuery errors
- SSL Secured
- No major spelling errors

Project Overview

KYC verified by Coinsult