Coinsult

# Advanced Manual Smart Contract Audit

**Project:** BullyPit
**Website:** https://bullypit.co/

| ● **Low-risk** | ● **Medium-risk** | ● **High-risk** |
|---|---|---|
| 3 low-risk code issues found | 0 medium-risk code issues found | 0 high-risk code issues found |

**Contract address**
0x946da76a6D2EF0d20ac5C9ba43E37F5146c5728D

## 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 | 0x2eff59ed6c7e7ec64325b2ebd09ff1012e8c9623 | 1,000,000,000,000 | 100.000% |

## Source code

Coinsult was commissioned by BullyPit to perform an audit based on the following smart contract:

https://bscscan.com/address/0x946da76a6d2ef0d20ac5c9ba43e37f5146c5728d#code

# Manual Code Review

## ● Low-risk

3 low-risk code issues found.
Could be fixed, will not bring problems.

- Contract contains Reentrancy vulnerabilities:
  Additional information: This combination increases risk of malicious intent. While it may be justified by some complex mechanics (e.g. rebase, reflections, buyback).
  More information: Slither

```
     function _transfer(
        address from,
        address to,
        uint256 amount
    ) private {
        require(from != address(0), "ERC20: transfer from the zero
address");
        require(to != address(0), "ERC20: transfer to the zero
address");
        require(amount > 0, "Transfer amount must be greater than
zero");
        if(from != owner() && to != owner())
            require(amount <= _maxTxAmount, "Transfer amount exceeds
the maxTxAmount.");

        // is the token balance of this contract address over the min
number of
        // tokens that we need to initiate a swap + liquidity lock?
        // also, don't get caught in a circular liquidity event.
        // also, don't swap & liquify if sender is uniswap pair.
        uint256 contractTokenBalance = balanceOf(address(this));

        if(contractTokenBalance >= _maxTxAmount)
        {
            contractTokenBalance = _maxTxAmount;
        }

        bool overMinTokenBalance = contractTokenBalance >=
numTokensSellToAddToLiquidity;
        if (
            overMinTokenBalance &&
            !inSwapAndLiquify &&
            from != uniswapV2Pair &&
            swapAndLiquifyEnabled
        ) {
            contractTokenBalance = numTokensSellToAddToLiquidity;
            //add liquidity
            swapAndLiquify(contractTokenBalance);
        }

        //indicates if fee should be deducted from transfer
        bool takeFee = true;

        //if any account belongs to _isExcludedFromFee account then
remove the fee
        if(_isExcludedFromFee[from] || _isExcludedFromFee[to]){
            takeFee = false;
        }

        //transfer amount, it will take tax, burn, liquidity fee
        _tokenTransfer(from,to,amount,takeFee);
    }
```

- Costly operations inside a loop might waste gas, so optimizations are justified.
  Use a local variable to hold the loop computation result.

```
function includeInReward(address account) external onlyOwner() {
        require(_isExcluded[account], "Account is already included");
        for (uint256 i = 0; i < _excluded.length; i++) {
            if (_excluded[i] == account) {
                _excluded[i] = _excluded[_excluded.length - 1];
                _tOwned[account] = 0;
                _isExcluded[account] = false;
                _excluded.pop();
                break;
            }
        }
    }
```

- Avoid relying on block.timestamp
  block.timestamp can be manipulated by miners.

```
function swapTokensForEth(uint256 tokenAmount) private {
        // generate the uniswap pair path of token -> weth
        address[] memory path = new address[](2);
        path[0] = address(this);
        path[1] = uniswapV2Router.WETH();

        _approve(address(this), address(uniswapV2Router), tokenAmount);

        // make the swap

uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
            tokenAmount,
            0, // accept any amount of ETH
            path,
            address(this),
            block.timestamp
        );
    }
```

## ● Medium-risk

0 medium-risk code issues found.
Should be fixed, could bring problems.

## ● High-risk

0 high-risk code issues found
Must be fixed, and will bring problems.

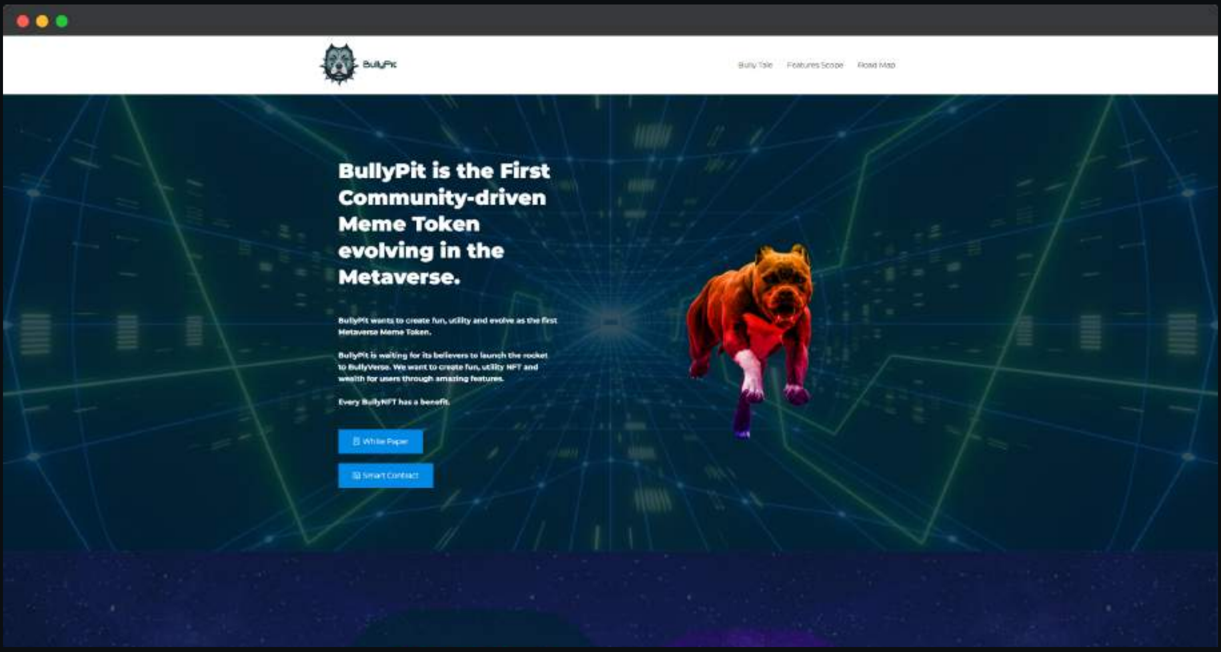
## Extra notes by the team

- Fees can be set up to 100% for both buy and sell fees.
- Dev notes can be deleted upon deployment
- The ownership of the contract isn’t renounced.
- Owner can whitelist addresses from fees.
- Owner can set a max transaction amount.

# Contract Snapshot

```
contract BullyPit is Context, IERC20, Ownable {
    using SafeMath for uint256;
    using Address for address;

    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;
    address[] private _excluded;

    address private _charityWalletAddress =
0x000000000000000000000000000000000000dEaD;

    uint256 private constant MAX = ~uint256(0);
    uint256 private _tTotal = 1000000 * 10**6 * 10**9;
    uint256 private _rTotal = (MAX - (MAX % _tTotal));
    uint256 private _tFeeTotal;

    string private _name = "BullyPit";
    string private _symbol = "BPT";
    uint8 private _decimals = 9;

    uint256 public _taxFee = 3;
    uint256 private _previousTaxFee = _taxFee;

    uint256 public _charityFee = 2;
    uint256 private _previousCharityFee = _charityFee;
    uint256 public _liquidityFee = 3;
    uint256 private _previousLiquidityFee = _liquidityFee;

    IUniswapV2Router02 public immutable uniswapV2Router;
    address public immutable uniswapV2Pair;
```

## Website Review



BullyPit is the First Community-driven Meme Token evolving in the Metaverse.

BullyPit wants to create fun, utility and evolve as the first Metaverse Meme Token.

BullyPit is waiting for its believers to launch the rocket to BullyVerse. We want to create fun, utility NFT and wealth for users through amazing features.

Every BullyNFT has a benefit.

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
● Contains no jQuery errors
● SSL Secured
● No major spelling errors

Loading speed: 85%

## Rug-pull Review

Based on the available information analyzed by us, we come to the following conclusions:

🟢 Locked Liquidity (no liquidity yet)

🟢 Large unlocked wallets

- Note: Tokens not distributed yet

🟢 Doxxed Team

## Honeypot Review

Based on the available information analyzed by us, we come to the following conclusions:

🟢 Ability to sell

🟢 Owner is not able to pause the contract

🟢 Router not hard coded in the contract

**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 the project owner.