

DESSERT
FINANCE



Birb (BIRB)

BEP-20 Audit

Performed at block 11996340

PERFORMED BY DESSERT FINANCE ON CONTRACT ADDRESSES:

0X82A479264B36104BE4FDB91618A59A4FC0F50650
0X0C13054755EFE54A0DCD1AAB3774513B119F85C9
0X1D37A555FE40C75CB8E8B05761C5A0777E0CC62F
0XB7DAA0F1CC90D23DE2C4C86A1B7F2B814BD62A8E
0X52181BE69892E3152D269247A6D2245E169EC174
0XB4256153222C2AA2CE56D13CE4A48E81499AB06B
0XC0F73D62F1137B7D9A7DC5D5EB2C7C5826C76189
0XCDFB966DE6F00089C6968622B450A2164FEE0800

INITIAL DISCLAIMER

Dessert Finance provides due-diligence project audits for various BSC projects. Dessert Finance in no way guarantees that a project will not remove liquidity, sell off team supply, or otherwise exit scam.

Dessert Finance does the legwork and provides public information about the project in an easy-to-understand format for the common person.

Agreeing to a project audit can be seen as a sign of confidence and is generally the first sign of trust for a project, but in no way guarantees that a team will not remove *all* liquidity (“Rug Pull”), sell off tokens, or completely exit scam. There is also no way to prevent private sale holders from selling off their tokens. It is ultimately your responsibility to read through all documentation, social media posts, and contract code of each individual project to draw your own conclusions and set your own risk tolerance.

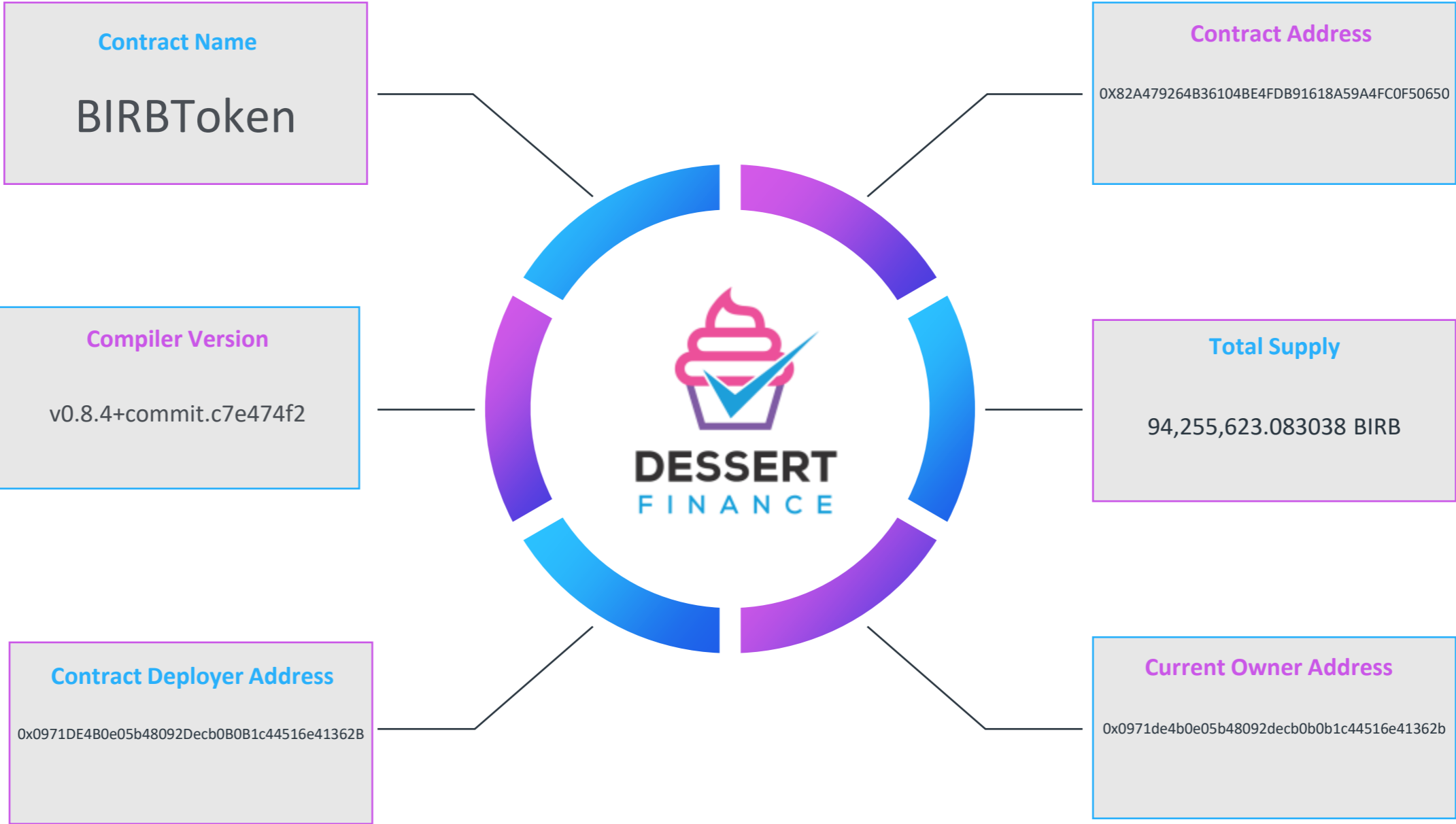
Dessert Finance in no way takes responsibility for any losses, nor does Dessert Finance encourage any speculative investments. The information provided in this audit is for information purposes only and should not be considered investment advice.

Table of Contents



1. Contract Code Audit – Token Overview
2. BEP-20 Contract Code Audit – Overview
3. BEP-20 Contract Code Audit – Vulnerabilities Checked
4. Contract Code Audit – Contract Ownership
5. Liquidity Ownership – Locked / Unlocked
6. Contract Code Audit – Mint Functions
7. Contract Transaction Fees
8. Website Overview
9. Social Media
10. Top Token Holders/Wallets
11. Location Audit
12. Review of Team
13. Roadmap
14. Disclaimers

Contract Code Audit – Token Overview



BEP-20 Contract Code Audit – Overview

Dessert Finance was commissioned to perform an audit on Birb (BIRB)

```
Submitted for verification at BscScan.com on 2021-07-08
//SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

//
//      .-----
//      |000011110000|
//      |000111110000|
//      |001111110000|
//      |011111110000|
//      |021111110000|
//      |031111110000|
//      |041111110000|
//      |051111110000|
//      |061111110000|
//      |071111110000|
//      |081111110000|
//      |091111110000|
//      |101111110000|
//      |111111110000|
//      |121111110000|
//      |131111110000|
//      |141111110000|
//      |151111110000|
//      |161111110000|
//      |171111110000|
//      |181111110000|
//      |191111110000|
//      |201111110000|
//      |211111110000|
//      |221111110000|
//      |231111110000|
//      |241111110000|
//      |251111110000|
//      |261111110000|
//      |271111110000|
//      |281111110000|
//      |291111110000|
//      |301111110000|
//      |311111110000|
//      |321111110000|
//      |331111110000|
//      |341111110000|
//      |351111110000|
//      |361111110000|
//      |371111110000|
//      |381111110000|
//      |391111110000|
//      |401111110000|
//      |411111110000|
//      |421111110000|
//      |431111110000|
//      |441111110000|
//      |451111110000|
//      |461111110000|
//      |471111110000|
//      |481111110000|
//      |491111110000|
//      |501111110000|
//      |511111110000|
//      |521111110000|
//      |531111110000|
//      |541111110000|
//      |551111110000|
//      |561111110000|
//      |571111110000|
//      |581111110000|
//      |591111110000|
//      |601111110000|
//      |611111110000|
//      |621111110000|
//      |631111110000|
//      |641111110000|
//      |651111110000|
//      |661111110000|
//      |671111110000|
//      |681111110000|
//      |691111110000|
//      |701111110000|
//      |711111110000|
//      |721111110000|
//      |731111110000|
//      |741111110000|
//      |751111110000|
//      |761111110000|
//      |771111110000|
//      |781111110000|
//      |791111110000|
//      |801111110000|
//      |811111110000|
//      |821111110000|
//      |831111110000|
//      |841111110000|
//      |851111110000|
//      |861111110000|
//      |871111110000|
//      |881111110000|
//      |891111110000|
//      |901111110000|
//      |911111110000|
//      |921111110000|
//      |931111110000|
//      |941111110000|
//      |951111110000|
//      |961111110000|
//      |971111110000|
//      |981111110000|
//      |991111110000|
//      |100111110000|
//
//      BIRB
//
/**
 * @dev Provides information about the current execution context, including the
 * sender of the transaction and its data. While these are generally available
 * via msg.sender and msg.data, they should not be accessed in such a direct
 * manner, since when dealing with EVM meta-transactions the account sending and
 * paying for execution may not be the actual sender (as far as application
 * logic is concerned).
 *
 * This contract is only required for intermediate, library-like contracts.
 */
abstract contract Context {
    function msgSender() internal view virtual returns (address) {
        return msg.sender;
    }

    function msgData() internal view virtual returns (bytes memory) {
        /**; // silence state mutability warning without generating bytecode
        return msg.data;
    }
}

/**
 * @dev Interface of the BEP20 standard as defined in the BEP.
 */
```

Contract Address

0x82A479264B36104be4FD9b91618a59A4fC0F50650

TokenTracker

Birb (BIRB)

Contract Creator

0x0971de4b0e05b48092decb0b0b1c44516e41362b

Source Code

Contract Source Code Verified

Contract Name

BIRBToken

Other Settings

default evmVersion, MIT

Compiler Version

v0.8.4+commit.c7e474f2

Optimization Enabled

Yes with 200 runs

Code is truncated to fit the constraints of this document.

[The code in its entirety can be viewed here.](#)

The contract code is **verified** on BSCScan.

BEP-20 Contract Code Audit – Vulnerabilities Checked

Vulnerability Tested	AI Scan	Human Review	Result
Integer Overflow	Complete	Complete	✓ Low / No Risk
Integer Underflow	Complete	Complete	✓ Low / No Risk
Correct Token Standards Implementation	Complete	Complete	✓ Low / No Risk
Timestamp Dependency for Crucial Functions	Complete	Complete	✓ Low / No Risk
Exposed _Transfer Function	Complete	Complete	✓ Low / No Risk
Transaction-Ordering Dependency	Complete	Complete	✓ Low / No Risk
Unchecked Call Return Variable	Complete	Complete	✓ Low / No Risk
Use of Deprecated Functions	Complete	Complete	✓ Low / No Risk
Unprotected SELFDESTRUCT Instruction	Complete	Complete	✓ Low / No Risk
State Variable Default Visibility	Complete	Complete	✓ Low / No Risk
Deployer Can Access User Funds	Complete	Complete	✓ Low / No Risk

The contract code is **verified** on BSCScan.

The vulnerabilities listed above were not found in the token's Smart Contract.

BEP-20 Contract Code Audit – Overview

Dessert Finance was commissioned to perform an audit on BirbFactory

```
Submitted for verification at BscScan.com on 2021-12-08
File: contracts/interfaces/BirbFactory.sol
pragma solidity >=0.5.0;

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

    function fee() external view returns (address);
    function feeToken() external view returns (address);

    function getPair(address token0, address token1) external view returns (address pair);
    function allPairs(uint) external view returns (address pair);
    function allPairsLength() external view returns (uint);

    function createPair(address token0, address token1) external returns (address pair);

    function setFee(address) external;
    function setFeeToken(address) external;
}

File: contracts/interfaces/BirbPair.sol
pragma solidity >=0.5.0;

interface IBirbPair {
    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 Mint(address indexed sender, uint amount0, uint amount1);
    event Burn(address indexed sender, uint amount0, uint amount1, address indexed to);
    event Swap(
        address indexed sender,
        uint amount0In,
        uint amount1In,
        uint amount0Out,
        uint amount1Out,
        address indexed to
    );
}
```

Contract Address

0x0c13054755efE54A0Dcd1Aab3774513b119F85c9

Contract Creator

0x714e3711974a68097ffaa211a4adbbe201ca40a1

Source Code

Contract Source Code Verified

Contract Name

BirbFactory

Other Settings

default evmVersion, None

Compiler Version

v0.5.16+commit.9c3226ce

Optimization Enabled

Yes with 200 runs

Code is truncated to fit the constraints of this document.

[The code in its entirety can be viewed here.](#)

The contract code is **verified** on BSCScan.

BEP-20 Contract Code Audit – Vulnerabilities Checked

Vulnerability Tested	AI Scan	Human Review	Result
Integer Overflow	Complete	Complete	✓ Low / No Risk
Integer Underflow	Complete	Complete	✓ Low / No Risk
Correct Token Standards Implementation	Complete	Complete	✓ Low / No Risk
Timestamp Dependency for Crucial Functions	Complete	Complete	✓ Low / No Risk
Exposed _Transfer Function	Complete	Complete	✓ Low / No Risk
Transaction-Ordering Dependency	Complete	Complete	✓ Low / No Risk
Unchecked Call Return Variable	Complete	Complete	✓ Low / No Risk
Use of Deprecated Functions	Complete	Complete	✓ Low / No Risk
Unprotected SELFDESTRUCT Instruction	Complete	Complete	✓ Low / No Risk
State Variable Default Visibility	Complete	Complete	✓ Low / No Risk
Deployer Can Access User Funds	Complete	Complete	✓ Low / No Risk

The contract code is **verified** on BSCScan.

The vulnerabilities listed above were not found in the token's Smart Contract.

BEP-20 Contract Code Audit – Overview

Dessert Finance was commissioned to perform an audit on BirbRouter

```
Submitted for verification at BscScan.com on 2021-10-04
// File: @binance/101/contracts/libraries/TransferHelper.sol
pragma solidity ^0.4.18;

// helper methods for interacting with ERC20 tokens and sending ETH that do not consistently return true/false
library TransferHelper {
    function safeApprove(address token, address to, uint value) internal {
        // bytes4(keccak256(bytes('approve(address,uint256)')));
        (bool success, bytes memory data) = token.call(abi.encodeWithSelector(keccak256("approve(address,uint256)")),
            to, value);
        require(success && (data.length == 0 || abi.decode(data, (bool))), "TransferHelper: APPROVE_FAILED");
    }

    function safeTransfer(address token, address to, uint value) internal {
        // bytes4(keccak256(bytes('transfer(address,uint256)')));
        (bool success, bytes memory data) = token.call(abi.encodeWithSelector(keccak256("transfer(address,uint256)")),
            to, value);
        require(success && (data.length == 0 || abi.decode(data, (bool))), "TransferHelper: TRANSFER_FAILED");
    }

    function safeTransferFrom(address token, address from, address to, uint value) internal {
        // bytes4(keccak256(bytes('transferFrom(address,address,uint256)')));
        (bool success, bytes memory data) = token.call(abi.encodeWithSelector(keccak256("transferFrom(address,address,uint256)")),
            from, to, value);
        require(success && (data.length == 0 || abi.decode(data, (bool))), "TransferHelper: TRANSFER_FROM_FAILED");
    }

    function safeTransferETH(address to, uint value) internal {
        (bool success, ) = to.call{value:value}("receive(bytes)");
        require(success, "TransferHelper: ETH_TRANSFER_FAILED");
    }
}

// File: contracts/interfaces/IBirbFactory.sol
pragma solidity ^0.4.18;

interface IBirbFactory {
    event PairCreated(address indexed tokenA, address indexed tokenB, address pair, uint);

    function fee() external view returns (address);
    function feeCollector() 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 addPaired(address) external;
    function addPairedTo(address) external;
}

// File: contracts/interfaces/IBirbRouter.sol
pragma solidity ^0.4.18;

interface IBirbRouter {
```

Contract Address

0x1D37a555Fe40C75Cb8E8B05761c5A0777E0CC62f

Contract Creator

0x714e3711974a68097ffaa211a4adbbe201ca40a1

Source Code

Contract Source Code Verified

Contract Name

BirbRouter

Other Settings

default evmVersion, None

Compiler Version

v0.6.6+commit.6c089d02

Optimization Enabled

Yes with 200 runs

Code is truncated to fit the constraints of this document.

[The code in its entirety can be viewed here.](#)

The contract code is **verified** on BSCScan.

BEP-20 Contract Code Audit – Vulnerabilities Checked

Vulnerability Tested	AI Scan	Human Review	Result
Integer Overflow	Complete	Complete	✓ Low / No Risk
Integer Underflow	Complete	Complete	✓ Low / No Risk
Correct Token Standards Implementation	Complete	Complete	✓ Low / No Risk
Timestamp Dependency for Crucial Functions	Complete	Complete	✓ Low / No Risk
Exposed _Transfer Function	Complete	Complete	✓ Low / No Risk
Transaction-Ordering Dependency	Complete	Complete	✓ Low / No Risk
Unchecked Call Return Variable	Complete	Complete	✓ Low / No Risk
Use of Deprecated Functions	Complete	Complete	✓ Low / No Risk
Unprotected SELFDESTRUCT Instruction	Complete	Complete	✓ Low / No Risk
State Variable Default Visibility	Complete	Complete	✓ Low / No Risk
Deployer Can Access User Funds	Complete	Complete	✓ Low / No Risk

The contract code is **verified** on BSCScan.

The vulnerabilities listed above were not found in the token's Smart Contract.

BEP-20 Contract Code Audit – Overview

Dessert Finance was commissioned to perform an audit on Diamond Birb (DBIRB)

```
Submitted for verification at BscScan.com on 2021-10-21
File: node_modules/@openzeppelin/contracts/token/ERC20/Contract.sol
pragma solidity ^0.6.0;

/**
 * @dev Provides information about the current execution context, including the
 * sender of the transaction and its data. While these are generally available
 * via msg.sender and msg.data, they would not be available in other contexts
 * such as mempool or other virtual machines like the account abstraction
 * or paying for gas fees may not be the actual sender (as far as an application
 * is concerned).
 *
 * This contract is only required for intermediate, library-like contracts.
 */
contract Contract {
    function msgSender() internal view returns (address payable) {
        return msg.sender;
    }

    function msgData() internal view returns (bytes memory) {
        return msg.data;
    }
}

File: @openzeppelin/contracts/access/Ownable.sol
pragma solidity ^0.6.0;

/**
 * @dev Contract module which provides a basic access control mechanism, where
 * there is an account (an owner) that can be granted exclusive access to
 * specific functions.
 *
 * By default, the owner account will be the one that deploys the contract. This
 * can later be changed with {transferOwnership}.
 *
 * This module is used through inheritance. It will have no effect on the
 * contract if it is not inherited.
 */
contract Ownable is Contract {
    address private _owner;

    event OwnershipTransferred(address indexed previousOwner, address indexed newOwner);

    /**
     * @dev Initializes the contract setting the deployer as the initial owner.
     */
    constructor () internal {
        address msgSender = msg.sender;
        _owner = msgSender;
        emit OwnershipTransferred(address(0), msgSender);
    }

    /**
     * @dev Returns the address of the current owner.
     */
    function owner() public view returns (address) {
        return _owner;
    }
}
```

Contract Address

0xB7dAA0F1cc90d23de2C4c86A1B7F2B814BD62a8E

TokenTracker

Diamond Birb (DBIRB)

Contract Creator

0xa5f2c301d5632d2f94ab9182bbcce5d65b41e490

Source Code

Contract Source Code Verified

Contract Name

DBirbToken

Other Settings

default evmVersion, None

Compiler Version

v0.6.12+commit.27d51765

Optimization Enabled

Yes with 200 runs

Code is truncated to fit the constraints of this document.

[The code in its entirety can be viewed here.](#)

The contract code is **verified** on BSCScan.

BEP-20 Contract Code Audit – Vulnerabilities Checked

Vulnerability Tested	AI Scan	Human Review	Result
Integer Overflow	Complete	Complete	✓ Low / No Risk
Integer Underflow	Complete	Complete	✓ Low / No Risk
Correct Token Standards Implementation	Complete	Complete	✓ Low / No Risk
Timestamp Dependency for Crucial Functions	Complete	Complete	✓ Low / No Risk
Exposed _Transfer Function	Complete	Complete	✓ Low / No Risk
Transaction-Ordering Dependency	Complete	Complete	✓ Low / No Risk
Unchecked Call Return Variable	Complete	Complete	✓ Low / No Risk
Use of Deprecated Functions	Complete	Complete	✓ Low / No Risk
Unprotected SELFDESTRUCT Instruction	Complete	Complete	✓ Low / No Risk
State Variable Default Visibility	Complete	Complete	✓ Low / No Risk
Deployer Can Access User Funds	Complete	Complete	✓ Low / No Risk

The contract code is **verified** on BSCScan.

The vulnerabilities listed above were not found in the token's Smart Contract.

BEP-20 Contract Code Audit – Overview

Dessert Finance was commissioned to perform an audit on SyrupBar Token (SYRUP)

```
Committee for verification at BscScan.com on 2021-08-04
File: code_modules/gpm/through/gpm-120/contracts/DBMContract.sol
pragma solidity ^0.6.0;

/**
 * @dev Provides information about the current execution context, including the
 * sender of the transaction and its data. While these are generally available
 * via msg.sender and msg.data, they should not be accessed in such a direct
 * manner, since when dealing with EVM data (transactions, this, sender and
 * paying for resolution may not be the actual sender (as far as an application
 * is concerned).
 *
 * This contract is only required for intermediate, library-like contracts.
 */
contract Context {
    /** @dev Internal constructor, to prevent people from accidentally deploying
     * an instance of this contract, which should not be used via inheritance.
     */
    constructor() internal {}

    function _msgSender() internal view returns (address payable) {
        return msg.sender;
    }

    function _msgData() internal view returns (bytes memory) {
        this; // silence state mutability warning without generating bytecode - see https://github.com/ethereum/solidity/issues/2691
        return msg.data;
    }
}

File: code_modules/gpm/through/gpm-120/contracts/Token/DBMToken.sol
pragma solidity ^0.6.0;

/**
 * @dev Contract module which provides a basic access control mechanism, where
 * there is an account (an owner) that can be granted exclusive access to
 * specific functions.
 *
 * By default, the owner account will be the one that deploys the contract. This
 * can later be changed with {transferOwnership}.
 *
 * This module is used through inheritance. It will not available the modifier
 * 'onlyOwner', which can be applied to your functions to restrict their use to
 * the owner.
 */
contract Ownable is Context {
    address private _owner;

    event OwnershipTransferred(address indexed previousOwner, address indexed newOwner);

    /**
     * @dev Initializes the contract setting the deployer as the initial owner.
     */
    constructor() internal {
        address msgSender = _msgSender();
        _owner = msgSender;
        emit OwnershipTransferred(address(0), msgSender);
    }

    /**
     * @dev Returns the address of the current owner.
     */
}

The contract code is truncated to fit the constraints of this document.
The code in its entirety can be viewed here.
```

Contract Address

0x52181be69892E3152d269247a6d2245E169Ec174

TokenTracker

SyrupBar Token (SYRUP)

Contract Creator

0x714e3711974a68097ffaa211a4adbbbe201ca40a1

Source Code

Contract Source Code Verified

Contract Name

DBirbSyrupBar

Other Settings

default evmVersion, None

Compiler Version

v0.6.12+commit.27d51765

Optimization Enabled

Yes with 200 runs

Code is truncated to fit the constraints of this document.

[The code in its entirety can be viewed here.](#)

BEP-20 Contract Code Audit – Vulnerabilities Checked

Vulnerability Tested	AI Scan	Human Review	Result
Integer Overflow	Complete	Complete	✓ Low / No Risk
Integer Underflow	Complete	Complete	✓ Low / No Risk
Correct Token Standards Implementation	Complete	Complete	✓ Low / No Risk
Timestamp Dependency for Crucial Functions	Complete	Complete	✓ Low / No Risk
Exposed _Transfer Function	Complete	Complete	✓ Low / No Risk
Transaction-Ordering Dependency	Complete	Complete	✓ Low / No Risk
Unchecked Call Return Variable	Complete	Complete	✓ Low / No Risk
Use of Deprecated Functions	Complete	Complete	✓ Low / No Risk
Unprotected SELFDESTRUCT Instruction	Complete	Complete	✓ Low / No Risk
State Variable Default Visibility	Complete	Complete	✓ Low / No Risk
Deployer Can Access User Funds	Complete	Complete	✓ Low / No Risk

The contract code is **verified** on BSCScan.

The vulnerabilities listed above were not found in the token's Smart Contract.

BEP-20 Contract Code Audit – Overview

Dessert Finance was commissioned to perform an audit on DBirbReferral

```
of for verification at BscScan.com on 2023-10-19.
@openzeppelin/contracts/v4.9.3/contract.sol

I1818y --0.5 | R.3.3 |

Provides information about the current execution context, including the
sender and msg.data. While these are generally available
since when dealing with EVM meta-transactions the direct sender and
the execution may not be the actual sender (as far as an application
concerns).

Contract is only required for intermediate, library like contracts.

contract Context {
    function sender() internal view virtual returns (address payable) {
        return msg.sender;
    }

    function msgData() internal view virtual returns (bytes memory) {
        // allow state mutability without generating bytecode - see https://github.com/OpenZeppelin/contracts/issues/112
        return msg.data;
    }
}

@openzeppelin/contracts/access/Ownable.sol

I1818y --0.5 | R.3.3 |

Contract module which provides a basic access control mechanism, where
only an account (an owner) can be granted exclusive access to
the functions.

By default, the owner account will be the one that deploys the contract. This
can be changed with {transferOwnership}.

Note: In order to use through inheritance, it will not be available the modifier
owner, which can be applied to your functions to restrict their use to
owner.

contract Ownable is Context {
    address private _owner;

    constructor(address indexed previousOwner, address indexed newOwner);

    // Initializes the contract setting the deployer as the initial owner.
    function () internal {
        address msgSender = msg.sender();
        _owner = msgSender;
        emit OwnershipTransferred(address(0), msgSender);
    }

    // Returns the address of the current owner.
    function owner() public view virtual returns (address) {
        return _owner;
    }
}
```

Contract Address

0xb4256153222c2aA2CE56D13CE4A48e81499ab06B

Contract Creator

0xa5f2c301d5632d2f94ab9182bbcce5d65b41e490

Source Code

Contract Source Code Verified

Contract Name

DBirbReferral

Other Settings

default evmVersion, None

Compiler Version

v0.6.12+commit.27d51765

Optimization Enabled

Yes with 200 runs

Code is truncated to fit the constraints of this document.

[The code in its entirety can be viewed here.](#)

The contract code is **verified** on BSCScan.

BEP-20 Contract Code Audit – Vulnerabilities Checked


Vulnerability Tested	AI Scan	Human Review	Result
Integer Overflow	Complete	Complete	✓ Low / No Risk
Integer Underflow	Complete	Complete	✓ Low / No Risk
Correct Token Standards Implementation	Complete	Complete	✓ Low / No Risk
Timestamp Dependency for Crucial Functions	Complete	Complete	✓ Low / No Risk
Exposed _Transfer Function	Complete	Complete	✓ Low / No Risk
Transaction-Ordering Dependency	Complete	Complete	✓ Low / No Risk
Unchecked Call Return Variable	Complete	Complete	✓ Low / No Risk
Use of Deprecated Functions	Complete	Complete	✓ Low / No Risk
Unprotected SELFDESTRUCT Instruction	Complete	Complete	✓ Low / No Risk
State Variable Default Visibility	Complete	Complete	✓ Low / No Risk
Deployer Can Access User Funds	Complete	Complete	✓ Low / No Risk

The contract code is **verified** on BSCScan.

The vulnerabilities listed above were not found in the token's Smart Contract.

BEP-20 Contract Code Audit – Overview

Dessert Finance was commissioned to perform an audit on DBirbMasterChef



```

// SPDX-License-Identifier: MIT
// File: @openzeppelin/contracts/math/SafeMath.sol

pragma solidity ^0.8.4;

/**
 * @dev Wrappers over Solidity's arithmetic operations with added overflow
 * checks.
 *
 * Arithmetic operations in Solidity wrap on overflow. This can easily result
 * in bugs, because programmers usually assume that an overflow raises an
 * error, which is the standard behavior in high-level programming languages.
 * `SafeMath` restores this intuition by reverting the transaction when an
 * operation overflows.
 *
 * Using this library instead of the unchecked operations eliminates an entire
 * class of bugs, so it's recommended to use it wherever.
 */
library SafeMath {
    /**
     * @dev Returns the addition of two unsigned integers, reverting on
     * overflow.
     *
     * Counterpart to Solidity's `+` operator.
     *
     * Requirements:
     * - Addition cannot overflow.
     */
    function add(uint256 a, uint256 b) internal pure returns (uint256) {
        uint256 c = a + b;
        require(c >= a, "SafeMath: addition overflow");

        return c;
    }

    /**
     * @dev Returns the subtraction of two unsigned integers, reverting on
     * overflow (when the result is negative).
     *
     * Counterpart to Solidity's `-` operator.
     *
     * Requirements:
     * - Subtraction cannot overflow.
     */
    function sub(uint256 a, uint256 b) internal pure returns (uint256) {
        return sub(a, b, "SafeMath: subtraction overflow");
    }

    /**
     * @dev Returns the subtraction of two unsigned integers, reverting with custom message on
     * overflow (when the result is negative).
     *
     * Counterpart to Solidity's `-` operator.
     *
     * Requirements:
     * - Subtraction cannot overflow.
     */
    function sub(uint256 a, uint256 b, string memory errorMessage) internal pure returns (uint256) {
        require(a >= b, errorMessage);

        return a - b;
    }
}

```

Contract Address

0xc0F73d62F1137B7d9A7DC5D5EB2c7C5826c76189

Contract Creator

0xa5f2c301d5632d2f94ab9182bbcce5d65b41e490

Source Code

Contract Source Code Verified

Contract Name

DBirbMasterChef

Other Settings

default evmVersion, None

Compiler Version

v0.6.12+commit.27d51765

Optimization Enabled

Yes with 200 runs

Code is truncated to fit the constraints of this document.

[The code in its entirety can be viewed here.](#)

The contract code is **verified** on BSCScan.

BEP-20 Contract Code Audit – Vulnerabilities Checked

Vulnerability Tested	AI Scan	Human Review	Result
Integer Overflow	Complete	Complete	✓ Low / No Risk
Integer Underflow	Complete	Complete	✓ Low / No Risk
Correct Token Standards Implementation	Complete	Complete	✓ Low / No Risk
Timestamp Dependency for Crucial Functions	Complete	Complete	✓ Low / No Risk
Exposed _Transfer Function	Complete	Complete	✓ Low / No Risk
Transaction-Ordering Dependency	Complete	Complete	✓ Low / No Risk
Unchecked Call Return Variable	Complete	Complete	✓ Low / No Risk
Use of Deprecated Functions	Complete	Complete	✓ Low / No Risk
Unprotected SELFDESTRUCT Instruction	Complete	Complete	✓ Low / No Risk
State Variable Default Visibility	Complete	Complete	✓ Low / No Risk
Deployer Can Access User Funds	Complete	Complete	✓ Low / No Risk

The contract code is **verified** on BSCScan.

The vulnerabilities listed above were not found in the token's Smart Contract.

BEP-20 Contract Code Audit – Overview

Dessert Finance was commissioned to perform an audit on DBirbVault



```
Submitted for verification at BscScan.com on 2021-06-20
...
Submitted for verification at BscScan.com on 2021-06-20
...
File: /home/.../contracts/contracts/DBirbVault.sol
SPM: 1.1.0 (max: 1.0) (min: 0.0)
pragma solidity ^0.6.12;

/**
 * @dev Provides information about the current execution context, including the
 * sender of the transaction and its data. While these are generally available
 * via msg.sender and msg.data, they should not be accessed in such a direct
 * manner, since when dealing with meta-transactions the account sending and
 * paying for execution may not be the actual sender (as far as an application
 * is concerned).
 *
 * This contract is only required for intermediate, library-like contracts.
 */
contract Context {
    function msgSender() internal view virtual returns (address payable) {
        return msg.sender;
    }

    function msgData() internal view virtual returns (bytes memory) {
        return msg.data;
    }
}

File: /home/.../contracts/contracts/DBirbVault.sol
SPM: 1.1.0 (max: 1.0) (min: 0.0)

/**
 * @dev Contract module which provides a basic access control mechanism, where
 * there is an account (or owner) that can be granted exclusive access to
 * specific functions.
 *
 * By default, the owner account will be the one that deploys the contract. This
 * can later be changed with {transferOwnership}.
 *
 * This module is used through inheritance. It will not compile if you inherit
 * from it, but you must inherit from its implementation to use its methods. Its
 * methods are marked as {abstract}.
 */
contract Ownable is Context {
    address private _owner;

    event OwnershipTransferred(address indexed previousOwner, address indexed newOwner);

    /**
     * @dev Initializes the contract setting the deployer as the initial owner.
     */
    constructor() internal {
        address msgSender = msgSender();
        _owner = msgSender();
        emit OwnershipTransferred(address(0), msgSender());
    }

    /**
     * @dev Returns the address of the current owner.
     */
    function owner() public view virtual returns (address) {
        return _owner;
    }
}
```

Contract Address

0x8a5F746032c128a487Dc19ec7245469E1995feDa

Contract Creator

0xa5f2c301d5632d2f94ab9182bbcce5d65b41e490

Source Code

Contract Source Code Verified

Contract Name

DBirbVault

Other Settings

default evmVersion, None

Compiler Version

v0.6.12+commit.27d51765

Optimization Enabled

Yes with 200 runs

Code is truncated to fit the constraints of this document.

[The code in its entirety can be viewed here.](#)

BEP-20 Contract Code Audit – Vulnerabilities Checked

Vulnerability Tested	AI Scan	Human Review	Result
Integer Overflow	Complete	Complete	✓ Low / No Risk
Integer Underflow	Complete	Complete	✓ Low / No Risk
Correct Token Standards Implementation	Complete	Complete	✓ Low / No Risk
Timestamp Dependency for Crucial Functions	Complete	Complete	✓ Low / No Risk
Exposed _Transfer Function	Complete	Complete	✓ Low / No Risk
Transaction-Ordering Dependency	Complete	Complete	✓ Low / No Risk
Unchecked Call Return Variable	Complete	Complete	✓ Low / No Risk
Use of Deprecated Functions	Complete	Complete	✓ Low / No Risk
Unprotected SELFDESTRUCT Instruction	Complete	Complete	✓ Low / No Risk
State Variable Default Visibility	Complete	Complete	✓ Low / No Risk
Deployer Can Access User Funds	Complete	Complete	✓ Low / No Risk

The contract code is **verified** on BSCScan.

The vulnerabilities listed above were not found in the token's Smart Contract.

BEP-20 Contract Code Audit – Vulnerabilities Checked

Vulnerability Tested	AI Scan	Human Review	Result
Integer Overflow	Complete	Complete	✓ Low / No Risk
Integer Underflow	Complete	Complete	✓ Low / No Risk
Correct Token Standards Implementation	Complete	Complete	✓ Low / No Risk
Timestamp Dependency for Crucial Functions	Complete	Complete	✓ Low / No Risk
Exposed _Transfer Function	Complete	Complete	✓ Low / No Risk
Transaction-Ordering Dependency	Complete	Complete	✓ Low / No Risk
Unchecked Call Return Variable	Complete	Complete	✓ Low / No Risk
Use of Deprecated Functions	Complete	Complete	✓ Low / No Risk
Unprotected SELFDESTRUCT Instruction	Complete	Complete	✓ Low / No Risk
State Variable Default Visibility	Complete	Complete	✓ Low / No Risk
Deployer Can Access User Funds	Complete	Complete	✓ Low / No Risk

The contract code is **verified** on BSCScan.

The vulnerabilities listed above were not found in the token's Smart Contract.

Contract Code Audit – Contract Ownership

Contract Ownership has not been renounced at the time of Audit



The contract ownership is not currently renounced.

We have placed the contract owner address below for your viewing:

[0x0971de4b0e05b48092decb0b0b1c44516e41362b](https://etherscan.io/address/0x0971de4b0e05b48092decb0b0b1c44516e41362b)

Liquidity Ownership – Locked / Unlocked

Locked liquidity information has been found.



This page will contain links to locked liquidity for the project if we are able to locate that information.

DxSale - Locked until July 23, 2023

https://dxsale.app/app/v2_9/dxlockview?id=0&add=0x0971DE4B0e05b48092Decb0B0B1c44516e41362B&type=lplock&chain=BSC

Contract Code Audit – Mint Functions

This Contract Cannot Mint New BIRB Tokens.



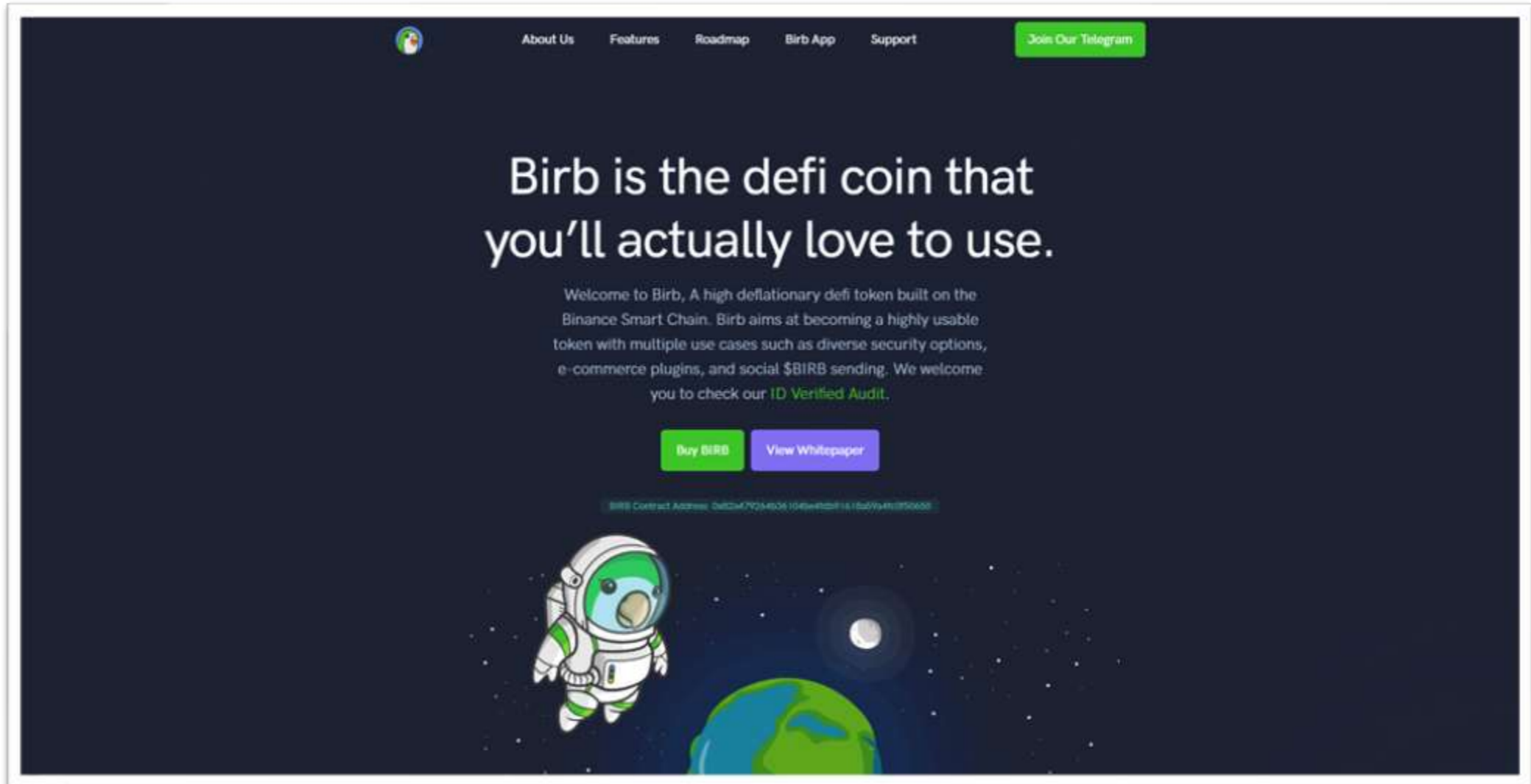
We do understand that sometimes mint functions are essential to the functionality of the project.

A mint function was not found in the contract code.

DBIRB tokens can only be minted by the MasterChef and not the Birb team.

Website Part 1 – Overview

www.birb.co



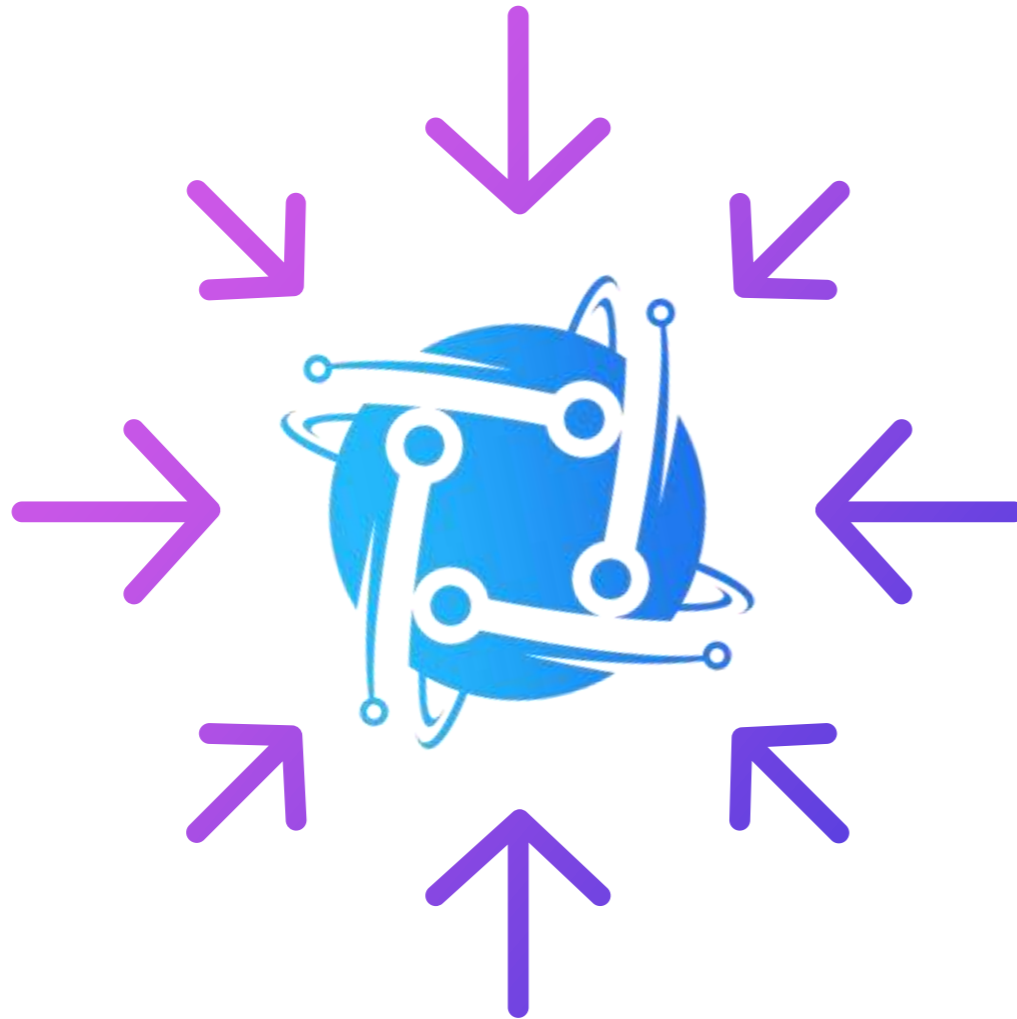
Above images are actual snapshots of the current live website of the project.

Website was registered on 03/28/2019, registration expires 03/28/2025.

✓ This exceeds the 3 year minimum we like to see on new projects.



Website Part 2 – Checklist



- ✓ Mobile Friendly
- ✓ No JavaScript Errors
- ✓ Spell Check
- ✓ SSL Certificate

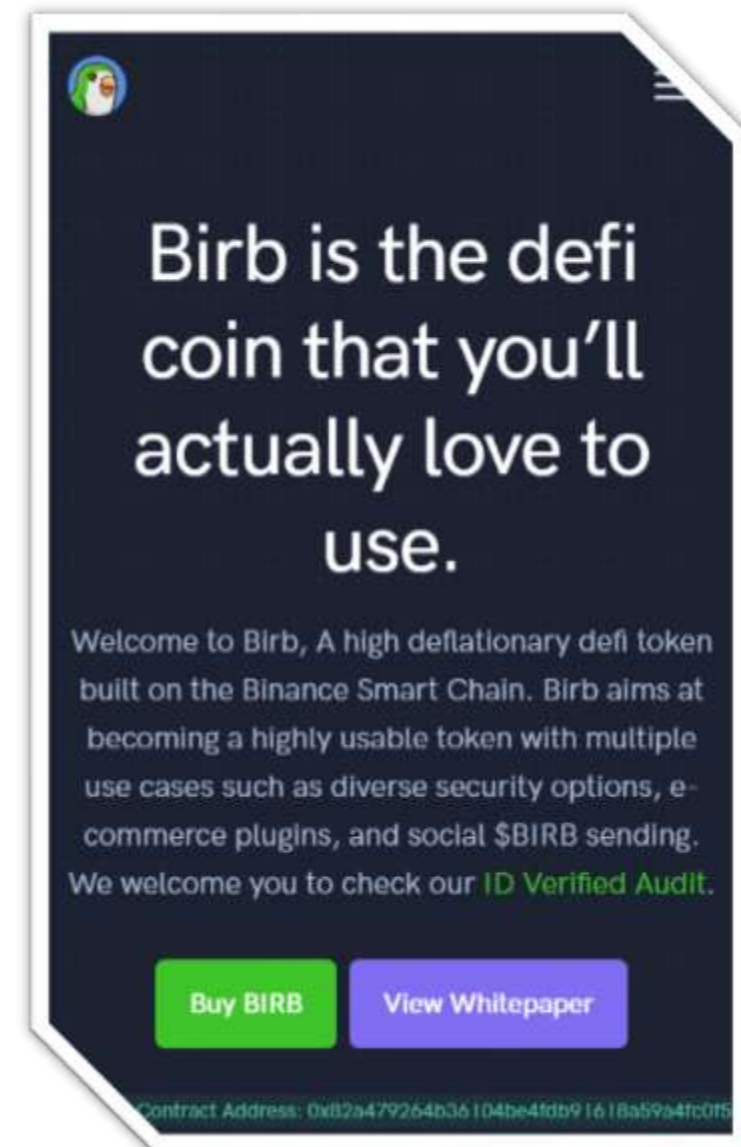
The website contained no JavaScript errors. No typos, or grammatical errors were present, and we found a valid SSL certificate allowing for access via https.

No additional issues were found on the website.

Website Part 3 – Responsive HTML5 & CSS3

No issues were found on the Mobile Friendly check for the website. All elements loaded properly and browser resize was not an issue. The team has put a considerable amount of thought and effort into making sure their website looks great on all screens.

No severe JavaScript errors were found. No issues with loading elements, code, or stylesheets.



Website Part 4 (GWS) – General Web Security



SSL CERTIFICATE

A valid SSL certificate was found. Details are as follows:

Offered to: Birb.co

Issued by: R3

Valid Until: 11/14/2021



CONTACT EMAIL

A valid contact email was found on the official website. Contact email is listed as shown below:

Contact

contact@birb.co



SPAM / MALWARE / POPUPS

No malware found

No injected spam found

No internal server errors

No popups found

Domain is marked clean by Google, McAfee, Sucuri Labs, & ESET



Social Media



We were able to locate a vast variety of Social Media networks.

All links have been conveniently placed below.



[Twitter](#)



[Telegram](#)



[Medium](#)



[Instagram](#)



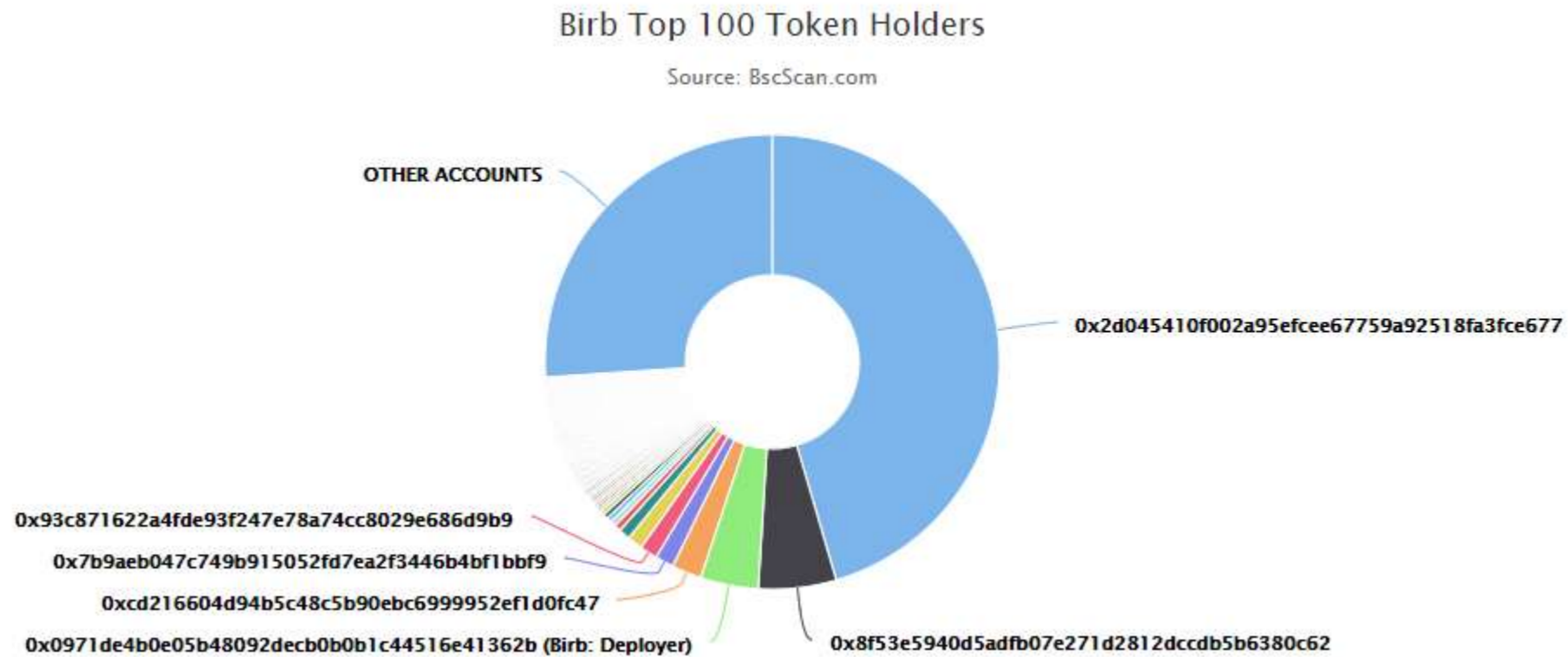
[YouTube](#)

✓ **At least 3 social media networks were found.**

Top Token Holders

The top token holders at the time of the audit are shown below.

[Click here to view the most up-to-date list of holders](#)



Rank	Address	Quantity (Token)	Percentage
1	0x2d045410f002a95efcee67759a92518fa3fce677	43,000,000	45.4690%
2	0x8f53e5940d5adfb07e271d2812dccdb5b6380c62	5,224,243.1	5.5242%
3	Birb: Deployer	3,893,537.6	4.1171%

Location Audit

The primary location of the Team is Texas, USA.



Team Overview

✓ Two founders have been DessertDoxxed with Government issued ID



Ankush Sharma

Blockchain + Full stack

Experience in blockchain development. Expert in frameworks including node.js, angular.js, react.js, vue.js, Codeigniter and Laravel.



Josue Sandoval

Web Application Developer

PuffedBirb is a front-end and back-end, web developer. UI/UX designer. I enjoy eCommerce plugin development. Yes, Jesse is my twin brother.



Jesse Sandoval

Web Application Developer

ChirpJedi is a front and back-end web and application developer. He is also an avid artist with experience in graphic design and SEO/SEM expert.



Armanto

Lead Artist

All of the beautiful Birb artwork you see on our website was hand-drawn and made into vector format by our talented artist, Armanto.



Linnea Houskeeper

Birb Writer, Proofreading

BA (Summa Cum Laude) in English Literature and Communications from Marywood University and MFA in Creative Writing from Lesley University.



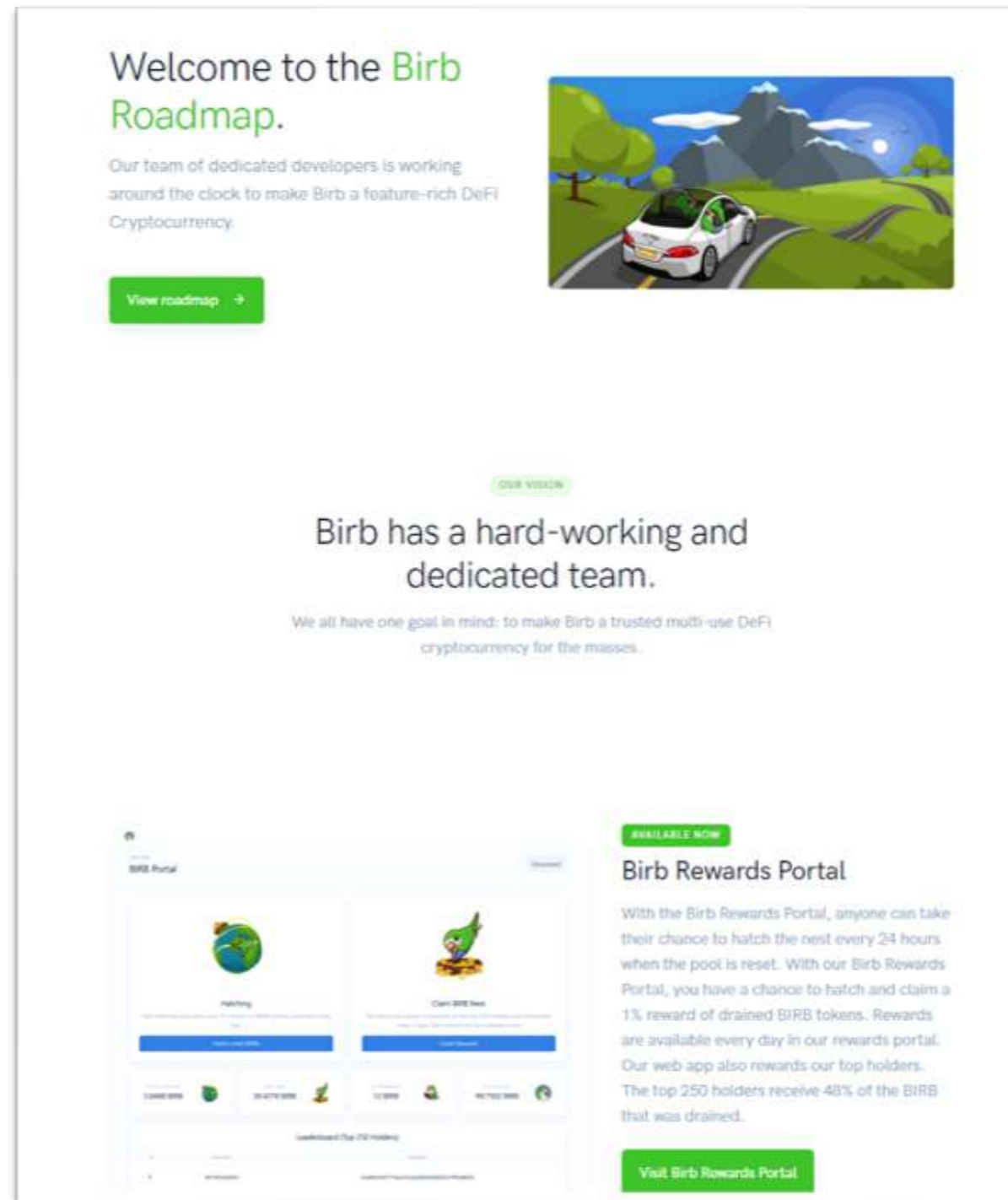
AnonBirb

Blockchain developer

An expert in blockchain and smart contract development. AnonBirb will be working on the privacy aspects of our project, including Flock Protocol.

Roadmap

A roadmap was found on the official website, we have conveniently placed it on this page for your viewing. [The full roadmap can be found here.](#)



The screenshot displays the Birb website's homepage. At the top left, it says "Welcome to the Birb Roadmap." followed by a paragraph: "Our team of dedicated developers is working around the clock to make Birb a feature-rich DeFi Cryptocurrency." Below this is a green button labeled "View roadmap →". To the right is an illustration of a white car on a winding road through a green landscape with mountains and a sun. In the center, under the heading "OUR VISION", it states "Birb has a hard-working and dedicated team." and "We all have one goal in mind: to make Birb a trusted multi-use DeFi cryptocurrency for the masses." At the bottom, there is a section for the "Birb Rewards Portal" with an "AVAILABLE NOW" tag. It includes a screenshot of the portal interface and a paragraph: "With the Birb Rewards Portal, anyone can take their chance to hatch the nest every 24 hours when the pool is reset. With our Birb Rewards Portal, you have a chance to hatch and claim a 1% reward of drained BIRB tokens. Rewards are available every day in our rewards portal. Our web app also rewards our top holders. The top 250 holders receive 48% of the BIRB that was drained." A green button "Visit Birb Rewards Portal" is located at the bottom right of this section.

Disclaimer



The opinions expressed in this document are for general informational purposes only and are **not intended to provide specific advice or recommendations for any individual or on any specific investment**. It is only intended to provide education and public knowledge regarding BSC projects. This audit is only applied to the type of auditing specified in this report and the scope of given in the results. Other unknown security vulnerabilities are beyond responsibility. Dessert Finance only issues this report based on the attacks or vulnerabilities that already existed or occurred before the issuance of this report. For the emergence of new attacks or vulnerabilities that exist or occur in the future, Dessert Finance lacks the capability to judge its possible impact on the security status of smart contracts, thus taking no responsibility for them. The smart contract analysis and other contents of this report are based solely on the documents and materials that the contract provider has provided to Dessert Finance or was publicly available before the issuance of this report (issuance of report recorded via block number on cover page), if the documents and materials provided by the contract provider are missing, tampered, deleted, concealed or reflected in a situation that is inconsistent with the actual situation, or if the documents and materials provided are changed after the issuance of this report, Dessert Finance assumes no responsibility for the resulting loss or adverse effects. Due to the technical limitations of any organization, this report conducted by Dessert Finance still has the possibility that the entire risk cannot be completely detected. Dessert Finance disclaims any liability for the resulting losses.

Dessert Finance provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Even projects with a low risk score have been known to pull liquidity, sell all team tokens, or exit-scam. Please exercise caution when dealing with any cryptocurrency related platforms.

The final interpretation of this statement belongs to Dessert Finance.

Dessert Finance highly advises against using cryptocurrencies as speculative investments and they should be used solely for the utility they aim to provide.



Thank You

DESSERT FINANCE PROJECT AUDIT HAS BEEN COMPLETED FOR BIRB (BIRB) AT BLOCK NUMBER: **11996340**
THIS AUDIT IS ONLY VALID IF VIEWED ON WWW.DSSERTSWAP.FINANCE

www.dessertswap.finance
<https://t.me/dessertswap>