DESSERT FINANCE

# Gravity (GVT)

BEP-20 Audit

Performed at block **25776363**

PERFORMED BY DESSERT FINANCE
FOR CONTRACT ADDRESS: 0x92BFE016A6352FE236e68Cb09619C1B38Ef96314, 0X84C507EF1B6B4D2C1DE9FF2737EE1609EA01E444, 0X1DE1A1205E8EAF08FDE233ECA5B5F41CBBC14925

## INITIAL DISCLAIMER

Dessert Finance provides due-diligence project audits for various 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 an audit in no way guarantees that a team will not remove ***all*** liquidity (“Rug Pull”), remove liquidity slowly, sell off tokens, quit the project, 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. Dessert Finance does not endorse, recommend, support, or suggest any projects that have been audited. An audit is an informational report based on our findings, We recommend you do your own research, we will never endorse any project to invest in.

# Table of Contents

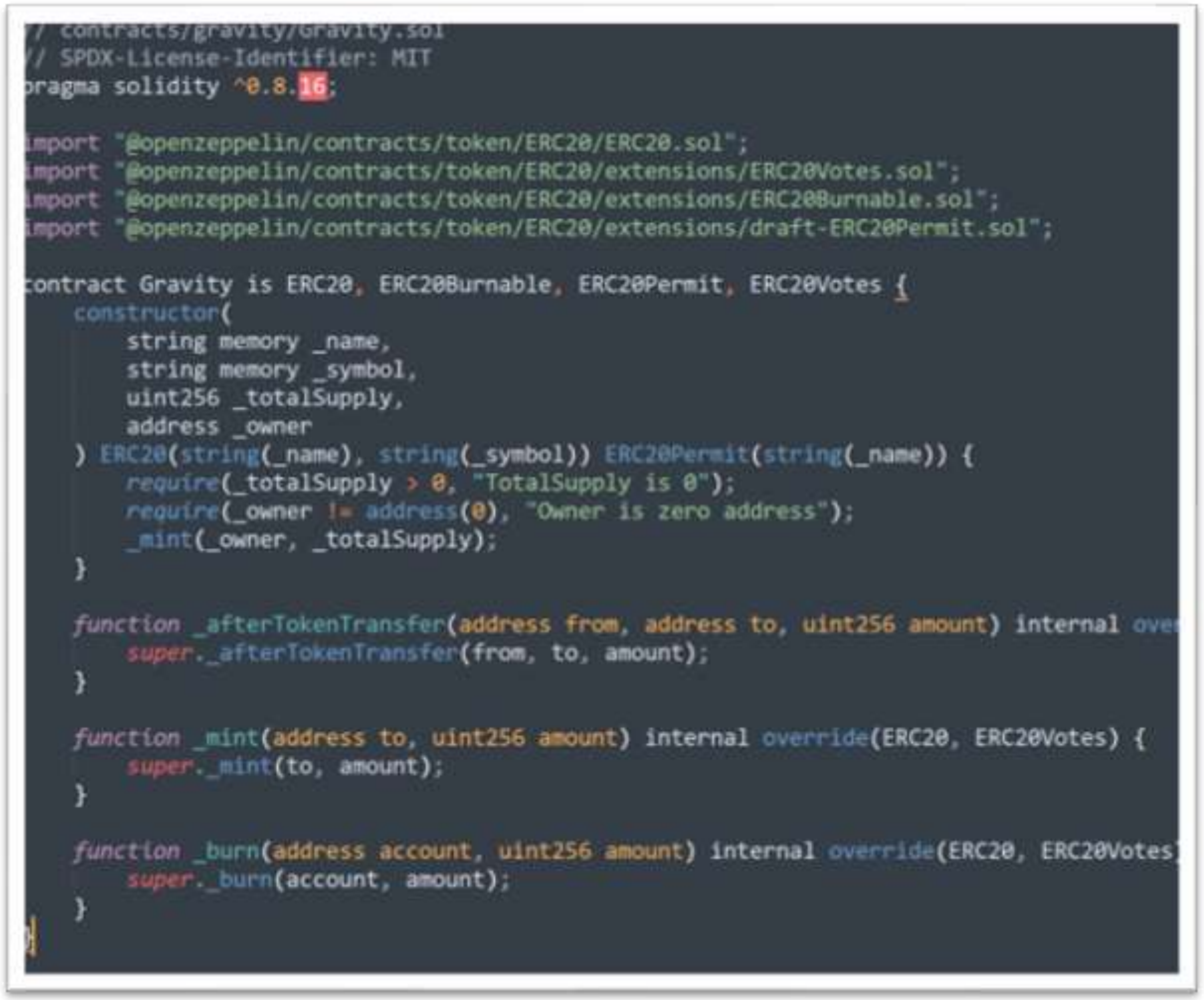
## Contract Code Audit – Token Overview

| | |
|---|---|
| Contract Name | Gravity |
| Contract Address | 0X92BFE016A6352FE236E68CB09619C1B38EF96314 |
| Compiler Version | v0.8.16+commit.07a7930e |
| Total Supply | 320,000,000 GVT |
| Contract Deployer Address | 0xdc6Ed9A7a2AFa95A2CB920FD50a14A75248E9561 |
| Decimals | 18 |

DESSERT FINANCE

## BEP-20 Contract Code Audit – Overview

Dessert Finance was commissioned to perform an audit on Gravity (GVT)



```
// contracts/gravity/Gravity.sol
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.16;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol";
import "@openzeppelin/contracts/token/ERC20/extensions/ERC20Burnable.sol";
import "@openzeppelin/contracts/token/ERC20/extensions/draft-ERC20Permit.sol";

contract Gravity is ERC20, ERC20Burnable, ERC20Permit, ERC20Votes {
    constructor(
        string memory _name,
        string memory _symbol,
        uint256 _totalSupply,
        address _owner
    ) ERC20(string(_name), string(_symbol)) ERC20Permit(string(_name)) {
        require(_totalSupply > 0, "TotalSupply is 0");
        require(_owner != address(0), "Owner is zero address");
        _mint(_owner, _totalSupply);
    }

    function _afterTokenTransfer(address from, address to, uint256 amount) internal ove
        super._afterTokenTransfer(from, to, amount);
    }

    function _mint(address to, uint256 amount) internal override(ERC20, ERC20Votes) {
        super._mint(to, amount);
    }

    function _burn(address account, uint256 amount) internal override(ERC20, ERC20Votes)
        super._burn(account, amount);
    }
}
```

**Contract Address**
0x92BFE016A6352FE236e68Cb09619C1B38Ef96314

**TokenTracker**
Gravity (GVT)

**Contract Creator**
0xdc6Ed9A7a2AFa95A2CB920FD50a14A75248E9561

**Source Code**
Contract Source Code Verified

**Contract Name**
Gravity

**Other Settings**
default evmVersion

**Compiler Version**
v0.8.16+commit.07a7930e

**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 |
|---|---|---|---|
| Compiler Errors | Complete | Complete | ✓ Low / No Risk |
| Outdated Compiler Version | Complete | Complete | ✓ Low / No Risk |
| 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. 0x92BFE016A6352FE236e68Cb09619C1B38Ef96314

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

## Contract Code Audit – Mint Functions

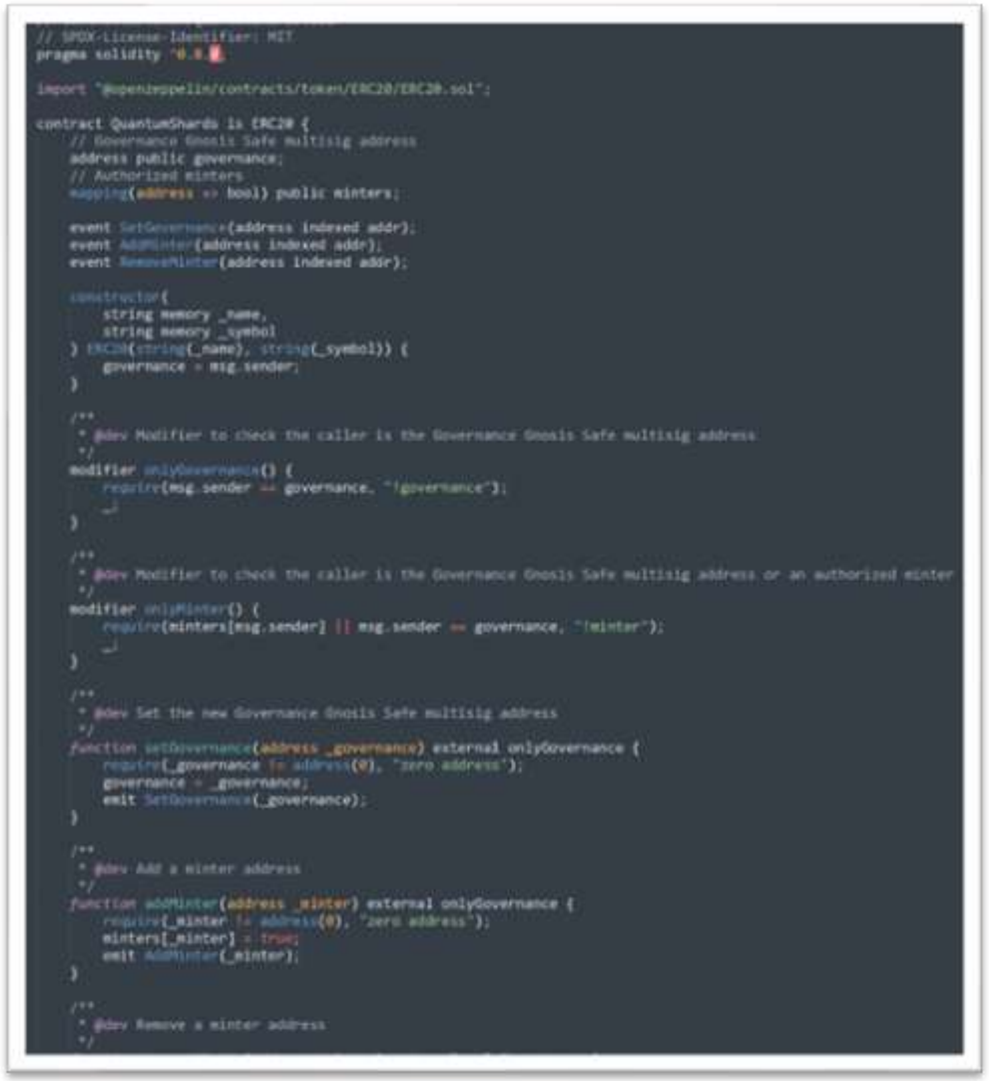
**This Contract Cannot Mint New GVT Tokens.**

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.**

## BEP-20 Contract Code Audit – Overview

Dessert Finance was commissioned to perform an audit on Quantum Shards (QTS)



```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract QuantumShards is ERC20 {
    // Governance Gnosis Safe multisig address
    address public governance;
    // Authorized minters
    mapping(address => bool) public minters;

    event SetGovernance(address indexed addr);
    event AddMinter(address indexed addr);
    event RemoveMinter(address indexed addr);

    constructor(
        string memory _name,
        string memory _symbol
    ) ERC20(string(_name), string(_symbol)) {
        governance = msg.sender;
    }

    /**
     * @dev Modifier to check the caller is the Governance Gnosis Safe multisig address
     */
    modifier onlyGovernance() {
        require(msg.sender == governance, "!governance");
        _;
    }

    /**
     * @dev Modifier to check the caller is the Governance Gnosis Safe multisig address or an authorized minter
     */
    modifier onlyMinter() {
        require(minters[msg.sender] || msg.sender == governance, "!minter");
        _;
    }

    /**
     * @dev Set the new Governance Gnosis Safe multisig address
     */
    function setGovernance(address _governance) external onlyGovernance {
        require(_governance != address(0), "zero address");
        governance = _governance;
        emit SetGovernance(_governance);
    }

    /**
     * @dev Add a minter address
     */
    function addMinter(address _minter) external onlyGovernance {
        require(_minter != address(0), "zero address");
        minters[_minter] = true;
        emit AddMinter(_minter);
    }

    /**
     * @dev Remove a minter address
     */
```

**Contract Address**
0x84c507Ef1B6b4D2c1dE9fF2737ee1609eA01e444

**TokenTracker**
Quantum Shards (QTS)

**Contract Creator**
0xdc6Ed9A7a2AFa95A2CB920FD50a14A75248E9561

**Source Code**
Contract Source Code Verified

**Contract Name**
QuantumShards

**Other Settings**
default evmVersion

**Compiler Version**
v0.8.3+commit.8d00100c

**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

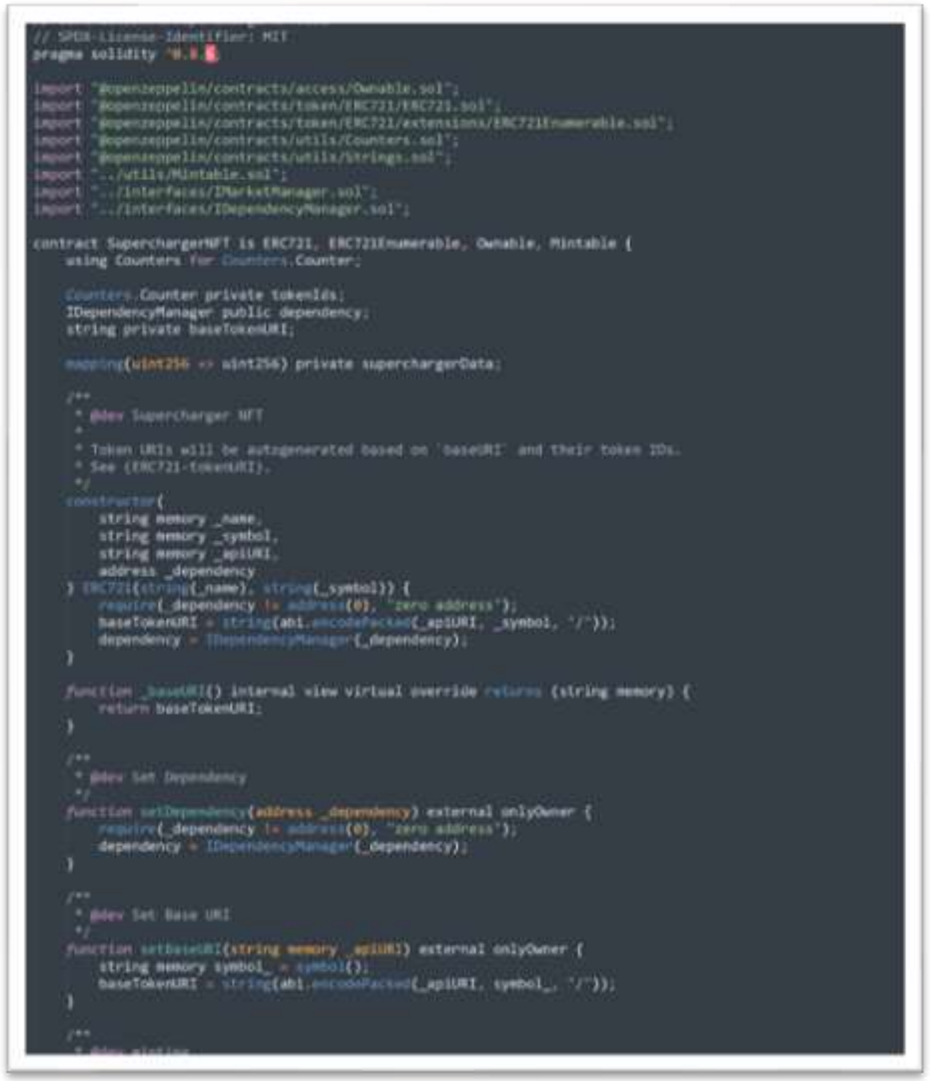
| Vulnerability Tested | AI Scan | Human Review | Result |
|---|---|---|---|
| Compiler Errors | Complete | Complete | ✓ Low / No Risk |
| Outdated Compiler Version | Complete | Complete | ✓ Low / No Risk |
| Integer Overflow | Complete | Complete | ✓ Low / No Risk |
| Integer Underflow | Complete | 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 |

QuantumShards

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 Supercharger NFT (SCHR)



```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.6;

import "@openzeppelin/contracts/access/Ownable.sol";
import "@openzeppelin/contracts/token/ERC721/ERC721.sol";
import "@openzeppelin/contracts/token/ERC721/extensions/ERC721Enumerable.sol";
import "@openzeppelin/contracts/utils/Counters.sol";
import "@openzeppelin/contracts/utils/Strings.sol";
import "../utils/Mintable.sol";
import "../interfaces/IMarketManager.sol";
import "../interfaces/IDependencyManager.sol";

contract SuperchargerNFT is ERC721, ERC721Enumerable, Ownable, Mintable {
    using Counters for Counters.Counter;

    Counters.Counter private tokenIds;
    IDependencyManager public dependency;
    string private baseTokenURI;

    mapping(uint256 => uint256) private superchargerData;

    /**
     * @dev Supercharger NFT
     *
     * Token URIs will be autogenerated based on `baseURI` and their token IDs.
     * See {ERC721-tokenURI}.
     */
    constructor(
        string memory _name,
        string memory _symbol,
        string memory _apiURI,
        address _dependency
    ) ERC721(string(_name), string(_symbol)) {
        require(_dependency != address(0), "zero address");
        baseTokenURI = string(abi.encodePacked(_apiURI, _symbol, "/"));
        dependency = IDependencyManager(_dependency);
    }

    function _baseURI() internal view virtual override returns (string memory) {
        return baseTokenURI;
    }

    /**
     * @dev Set Dependency
     */
    function setDependency(address _dependency) external onlyOwner {
        require(_dependency != address(0), "zero address");
        dependency = IDependencyManager(_dependency);
    }

    /**
     * @dev Set Base URI
     */
    function setbaseURI(string memory _apiURI) external onlyOwner {
        string memory symbol_ = symbol();
        baseTokenURI = string(abi.encodePacked(_apiURI, symbol_, "/"));
    }

    /**
     * @dev minting
```

**Contract Address**
0x1de1a1205e8EAF08fDE233ecA5B5f41cBbc14925

**TokenTracker**
Supercharger NFT (SCHR)

**Contract Creator**
0xdc6Ed9A7a2AFa95A2CB920FD50a14A75248E9561

**Source Code**
Contract Source Code Verified

**Contract Name**
SuperchargerNFT

**Other Settings**
**default** evmVersion

**Compiler Version**
v0.8.6+commit.11564f7e

**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

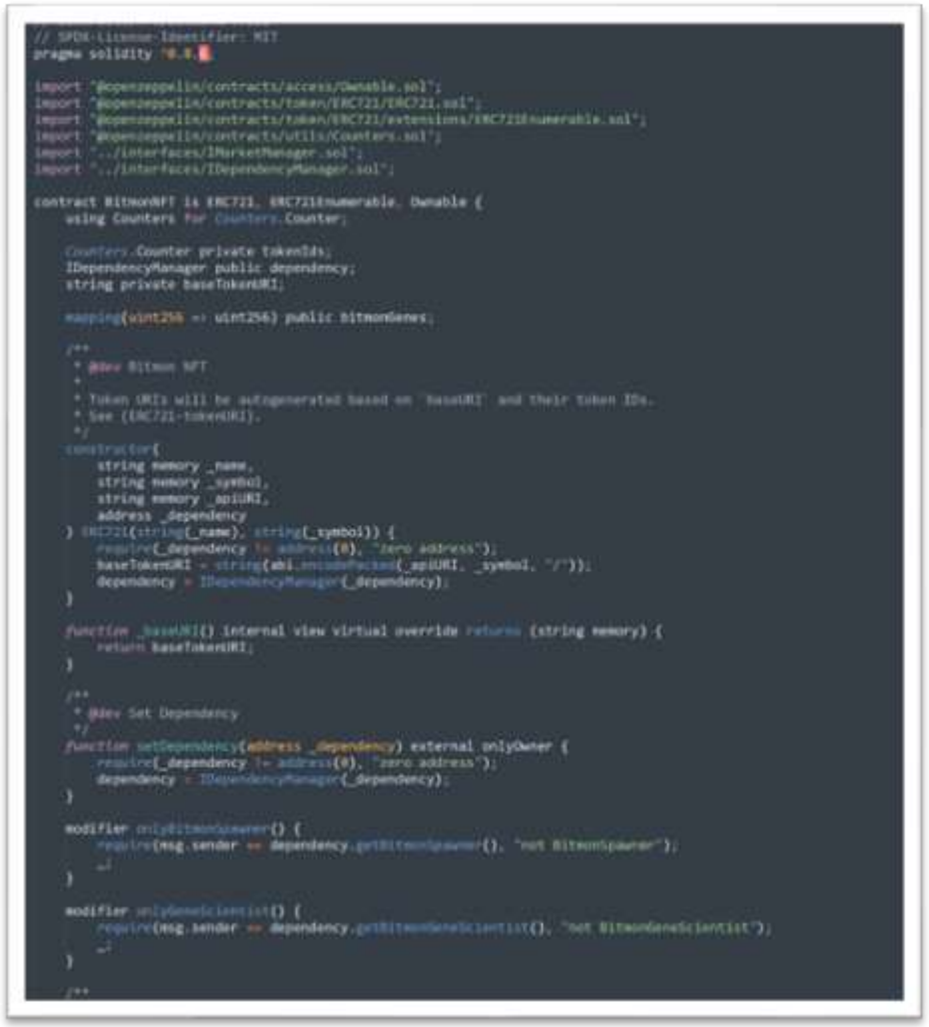
| Vulnerability Tested | AI Scan | Human Review | Result |
|---|---|---|---|
| Compiler Errors | Complete | Complete | ✓ Low / No Risk |
| Outdated Compiler Version | Complete | Complete | ✓ Low / No Risk |
| Integer Overflow | Complete | Complete | ✓ Low / No Risk |
| Integer Underflow | Complete | 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 |

SuperchargerNFT

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 Bitmon NFT (BMON)



```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.6;

import "@openzeppelin/contracts/access/Ownable.sol";
import "@openzeppelin/contracts/token/ERC721/ERC721.sol";
import "@openzeppelin/contracts/token/ERC721/extensions/ERC721Enumerable.sol";
import "@openzeppelin/contracts/utils/Counters.sol";
import "../interfaces/IMarketManager.sol";
import "../interfaces/IDependencyManager.sol";

contract BitmonNFT is ERC721, ERC721Enumerable, Ownable {
    using Counters for Counters.Counter;

    Counters.Counter private tokenIds;
    IDependencyManager public dependency;
    string private baseTokenURI;

    mapping(uint256 => uint256) public bitmonGenes;

    /**
     * @dev Bitmon NFT
     *
     * Token URIs will be autogenerated based on `baseURI` and their token IDs.
     * See {ERC721-tokenURI}.
     */
    constructor(
        string memory _name,
        string memory _symbol,
        string memory _apiURI,
        address _dependency
    ) ERC721(string(_name), string(_symbol)) {
        require(_dependency != address(0), "zero address");
        baseTokenURI = string(abi.encodePacked(_apiURI, _symbol, "/"));
        dependency = IDependencyManager(_dependency);
    }

    function _baseURI() internal view virtual override returns (string memory) {
        return baseTokenURI;
    }

    /**
     * @dev Set Dependency
     */
    function setDependency(address _dependency) external onlyOwner {
        require(_dependency != address(0), "zero address");
        dependency = IDependencyManager(_dependency);
    }

    modifier onlyBitmonSpawner() {
        require(msg.sender == dependency.getBitmonSpawner(), "not BitmonSpawner");
        _;
    }

    modifier onlyGeneScientist() {
        require(msg.sender == dependency.getBitmonGeneScientist(), "not BitmonGeneScientist");
        _;
    }

    /**
```

**Contract Address**
0xa9297D069D2d4453491E75E3c765EBF63d116d5C

**TokenTracker**
Bitmon NFT (BMON)

**Contract Creator**
0xdc6Ed9A7a2AFa95A2CB920FD50a14A75248E9561

**Source Code**
Contract Source Code Verified

**Contract Name**
BitmonNFT

**Other Settings**
**default** evmVersion

**Compiler Version**
v0.8.6+commit.11564f7e

**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 |
|---|---|---|---|
| Compiler Errors | Complete | Complete | ✓ Low / No Risk |
| Outdated Compiler Version | Complete | Complete | ✓ Low / No Risk |
| 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 |

BitmonNFT

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

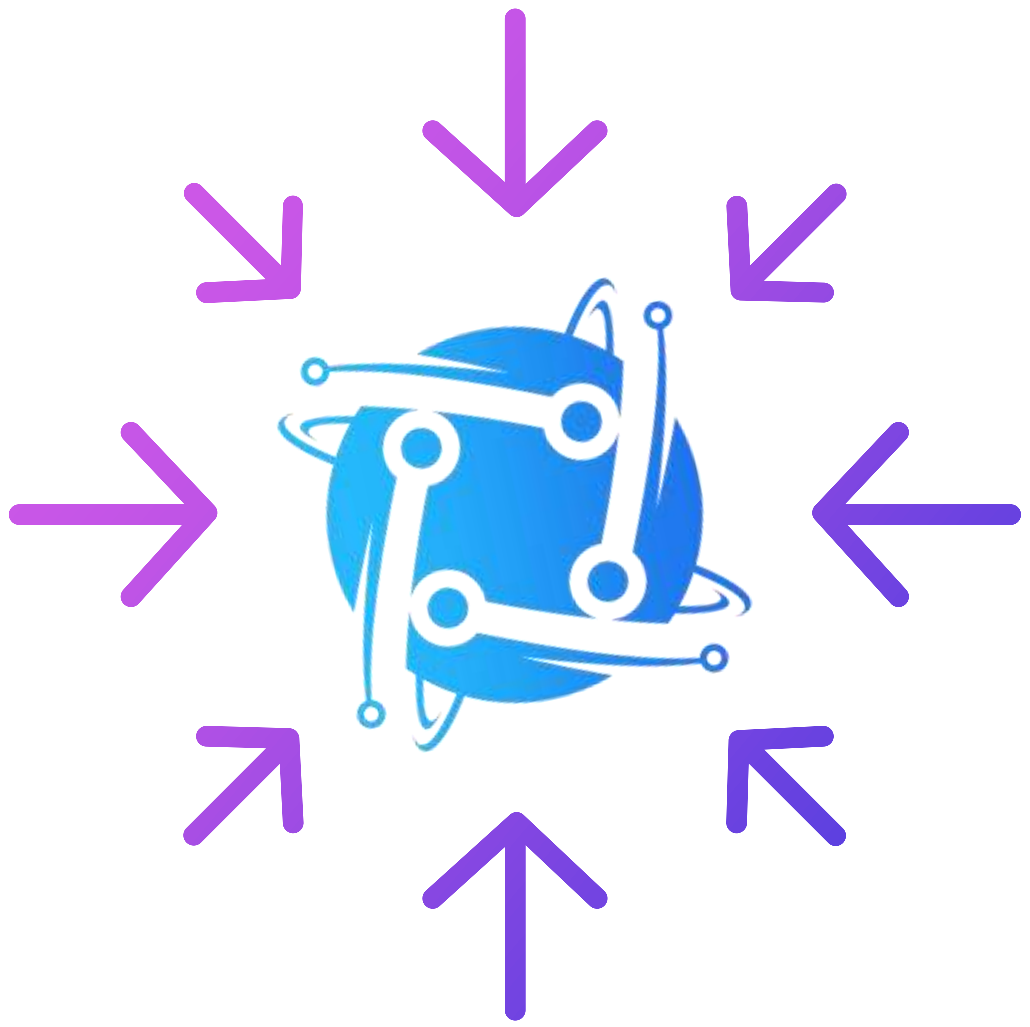
## Website Part 1 – Overview
[www.bitmonparadise.com](www.bitmonparadise.com)

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

Website was registered on 12/04/21, registration expires 12/04/2024.

✓ This meets 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: bitmonparaise.com

Issued by: cPanel, Inc

Valid Until: 06/11/2022

✓

**CONTACT EMAIL**

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

Contact

**support@bitmonparadise.com**

✓

**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 variety of Social Media networks for the project.

All links have been conveniently placed below.

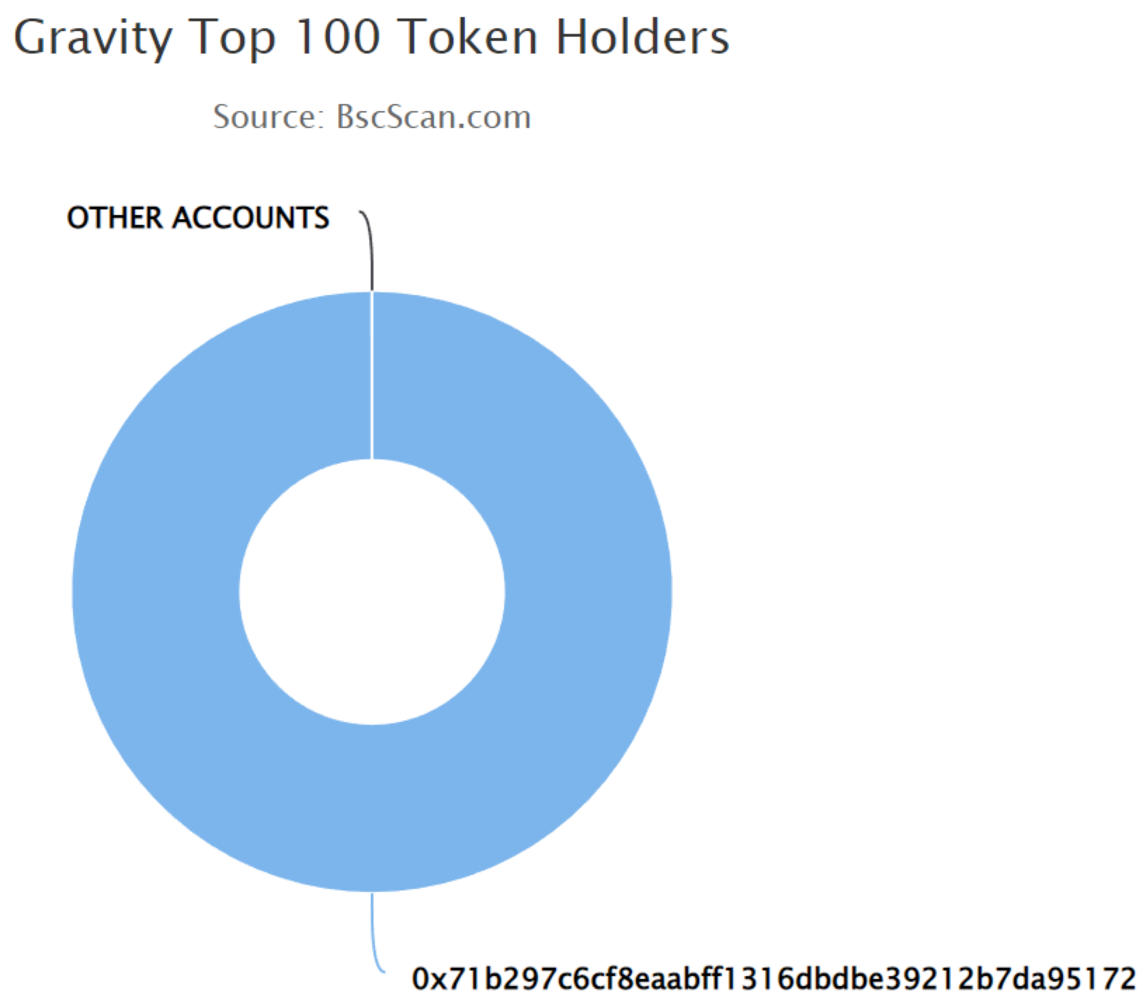
Twitter Telegram Reddit Discord Facebook Instagram Tiktok

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

## Top Token Holders

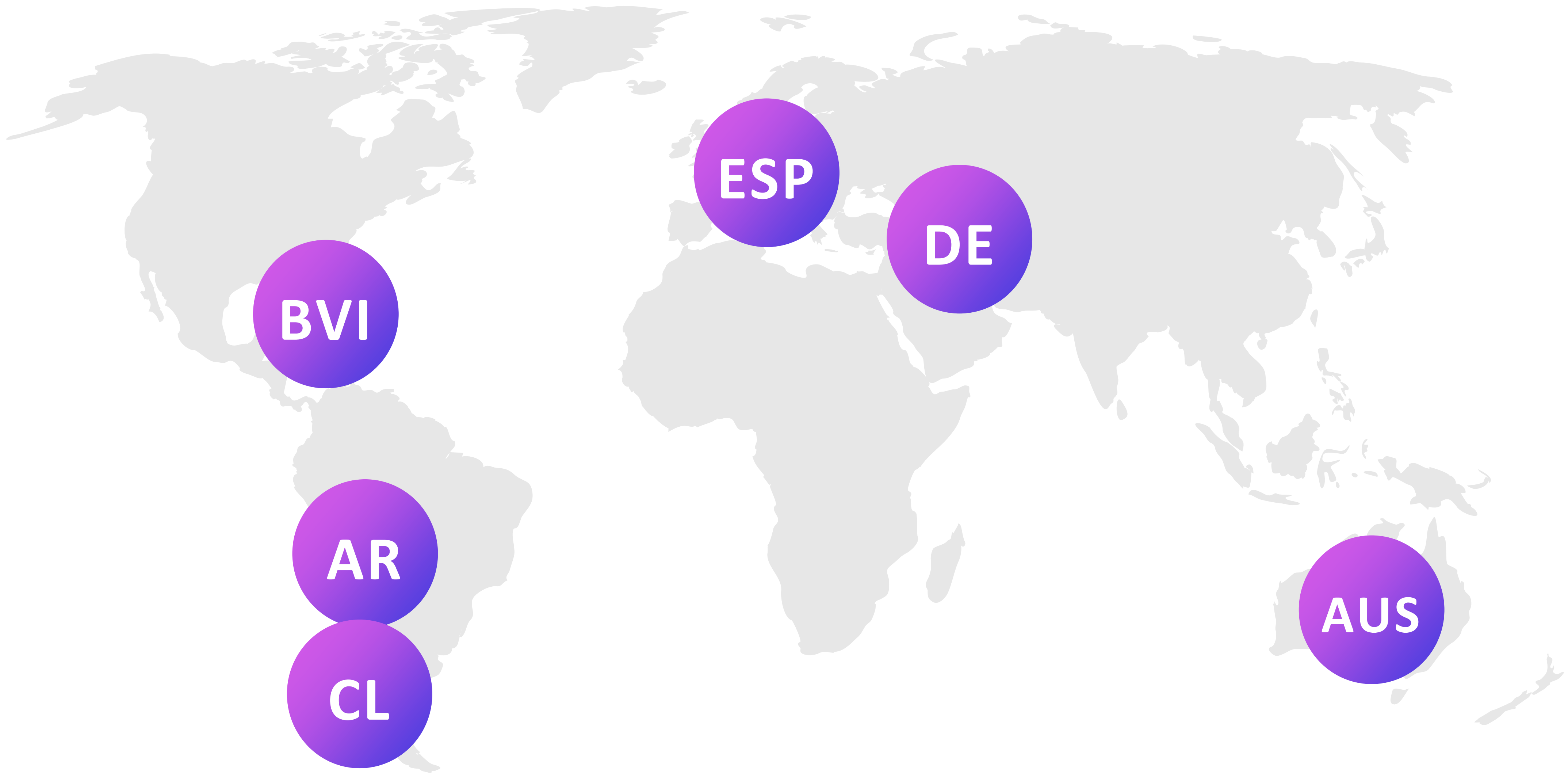
The entire supply was in one wallet at the time of audit. We expect this to change as the project goes through initial distribution phases. Please use the link below to view the most up-to-date holder information.

Click here to view the most up-to-date list of holders



## Location Audit

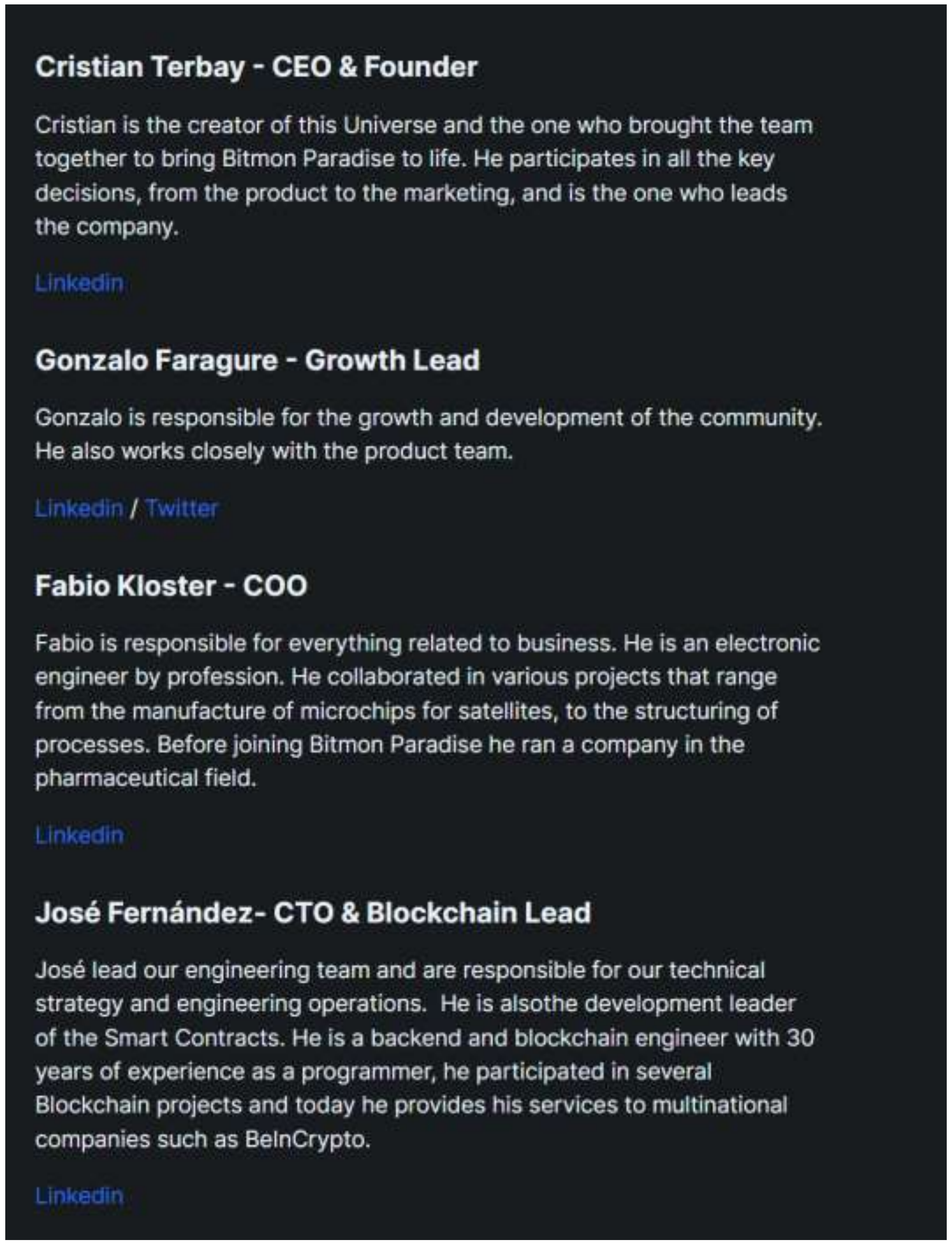
The company was found to be registered in the British Virgin Islands with team members from across the globe shown below.



# Team Overview

Team information was found on the projects website and is shown below for your convenience.

Please visit this page for direct links to team members social profiles.



**Cristian Terbay - CEO & Founder**

Cristian is the creator of this Universe and the one who brought the team together to bring Bitmon Paradise to life. He participates in all the key decisions, from the product to the marketing, and is the one who leads the company.

Linkedin

**Gonzalo Faragure - Growth Lead**

Gonzalo is responsible for the growth and development of the community. He also works closely with the product team.

Linkedin / Twitter

**Fabio Kloster - COO**

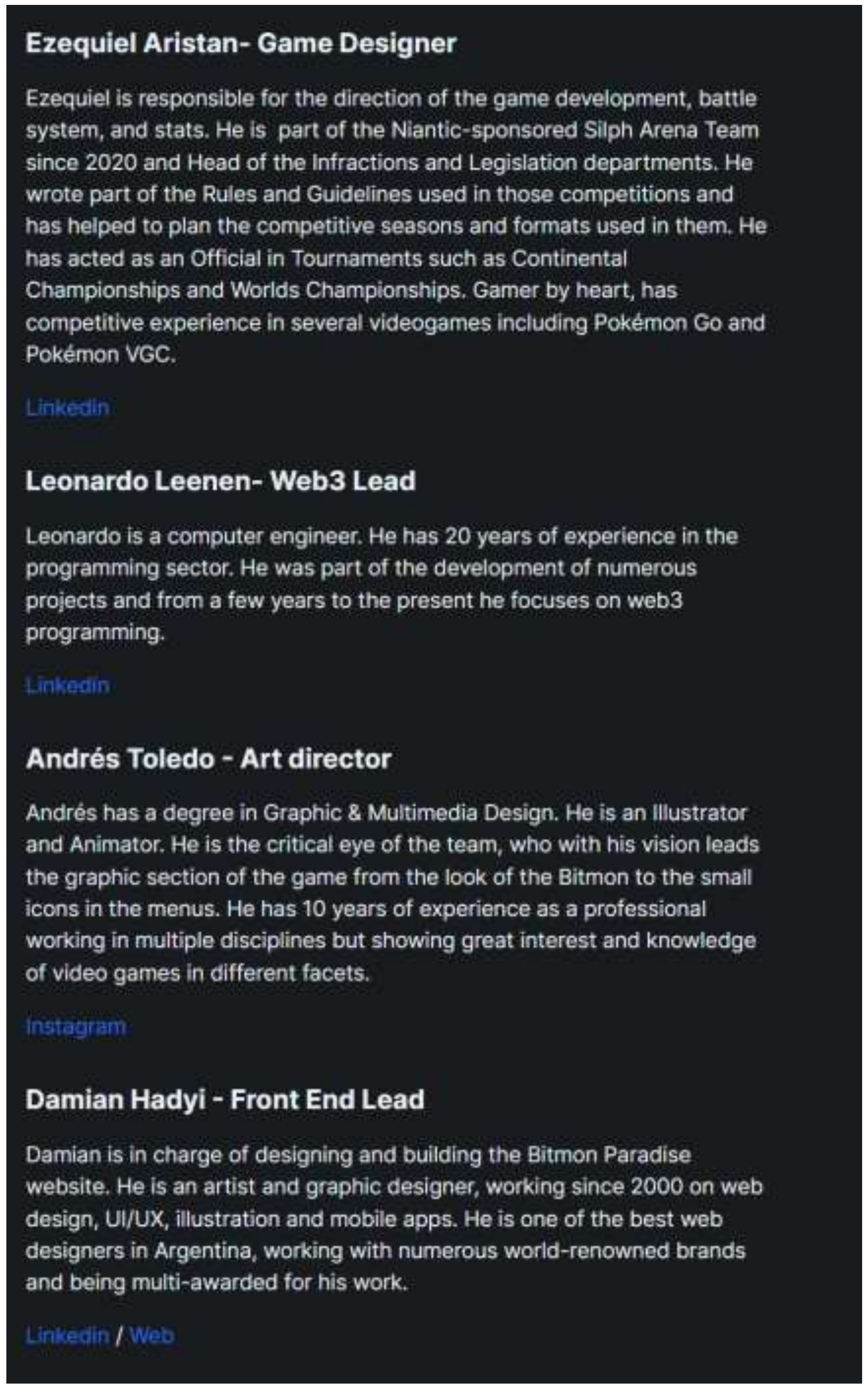
Fabio is responsible for everything related to business. He is an electronic engineer by profession. He collaborated in various projects that range from the manufacture of microchips for satellites, to the structuring of processes. Before joining Bitmon Paradise he ran a company in the pharmaceutical field.

Linkedin

**José Fernández- CTO & Blockchain Lead**

José lead our engineering team and are responsible for our technical strategy and engineering operations. He is alsothe development leader of the Smart Contracts. He is a backend and blockchain engineer with 30 years of experience as a programmer, he participated in several Blockchain projects and today he provides his services to multinational companies such as BeInCrypto.

Linkedin



**Ezequiel Aristan- Game Designer**

Ezequiel is responsible for the direction of the game development, battle system, and stats. He is part of the Niantic-sponsored Silph Arena Team since 2020 and Head of the Infractions and Legislation departments. He wrote part of the Rules and Guidelines used in those competitions and has helped to plan the competitive seasons and formats used in them. He has acted as an Official in Tournaments such as Continental Championships and Worlds Championships. Gamer by heart, has competitive experience in several videogames including Pokémon Go and Pokémon VGC.

Linkedin

**Leonardo Leenen- Web3 Lead**

Leonardo is a computer engineer. He has 20 years of experience in the programming sector. He was part of the development of numerous projects and from a few years to the present he focuses on web3 programming.

Linkedin

**Andrés Toledo - Art director**

Andrés has a degree in Graphic & Multimedia Design. He is an Illustrator and Animator. He is the critical eye of the team, who with his vision leads the graphic section of the game from the look of the Bitmon to the small icons in the menus. He has 10 years of experience as a professional working in multiple disciplines but showing great interest and knowledge of video games in different facets.

Instagram

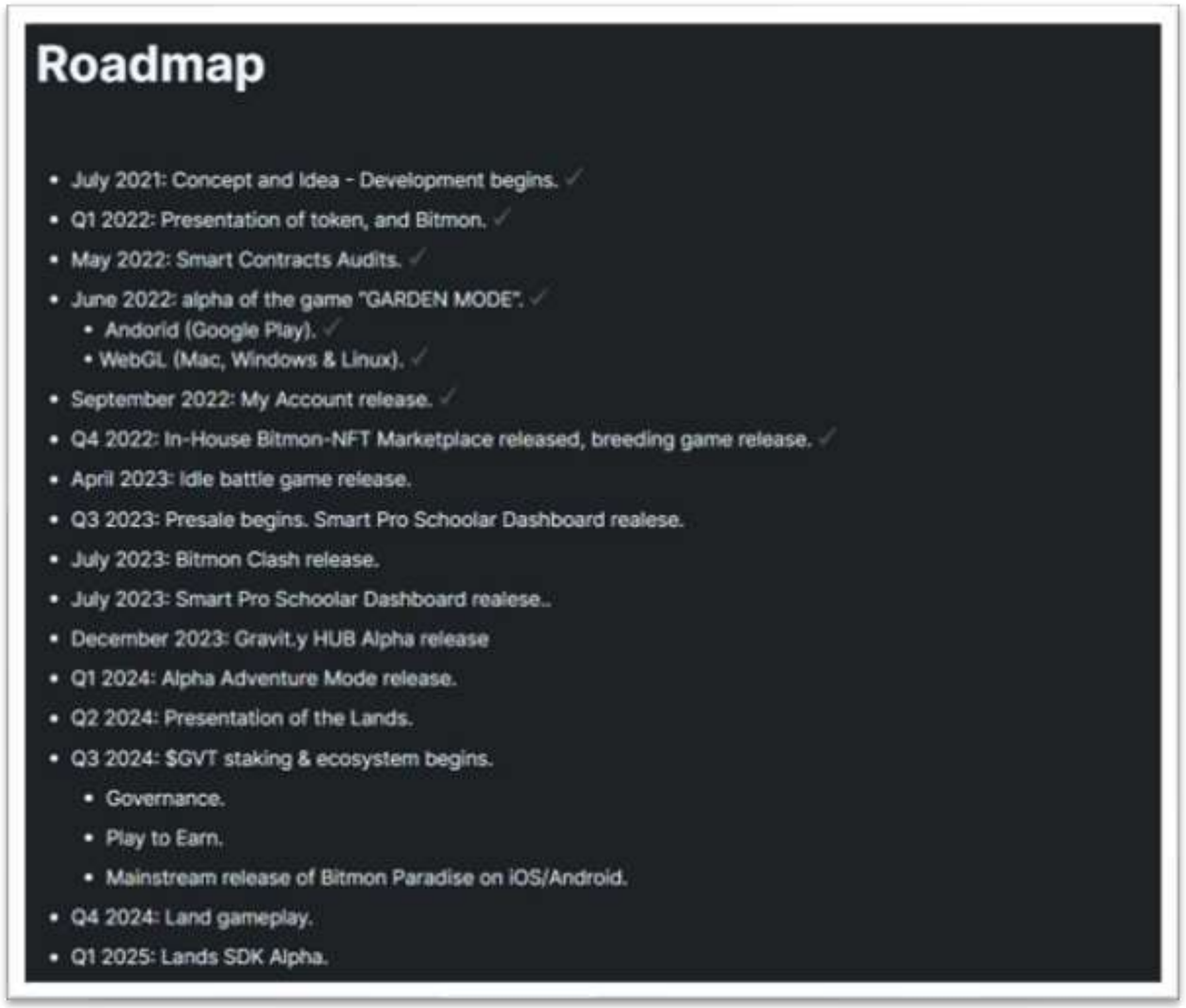
**Damian Hadyi - Front End Lead**

Damian is in charge of designing and building the Bitmon Paradise website. He is an artist and graphic designer, working since 2000 on web design, UI/UX, illustration and mobile apps. He is one of the best web designers in Argentina, working with numerous world-renowned brands and being multi-awarded for his work.

Linkedin / Web

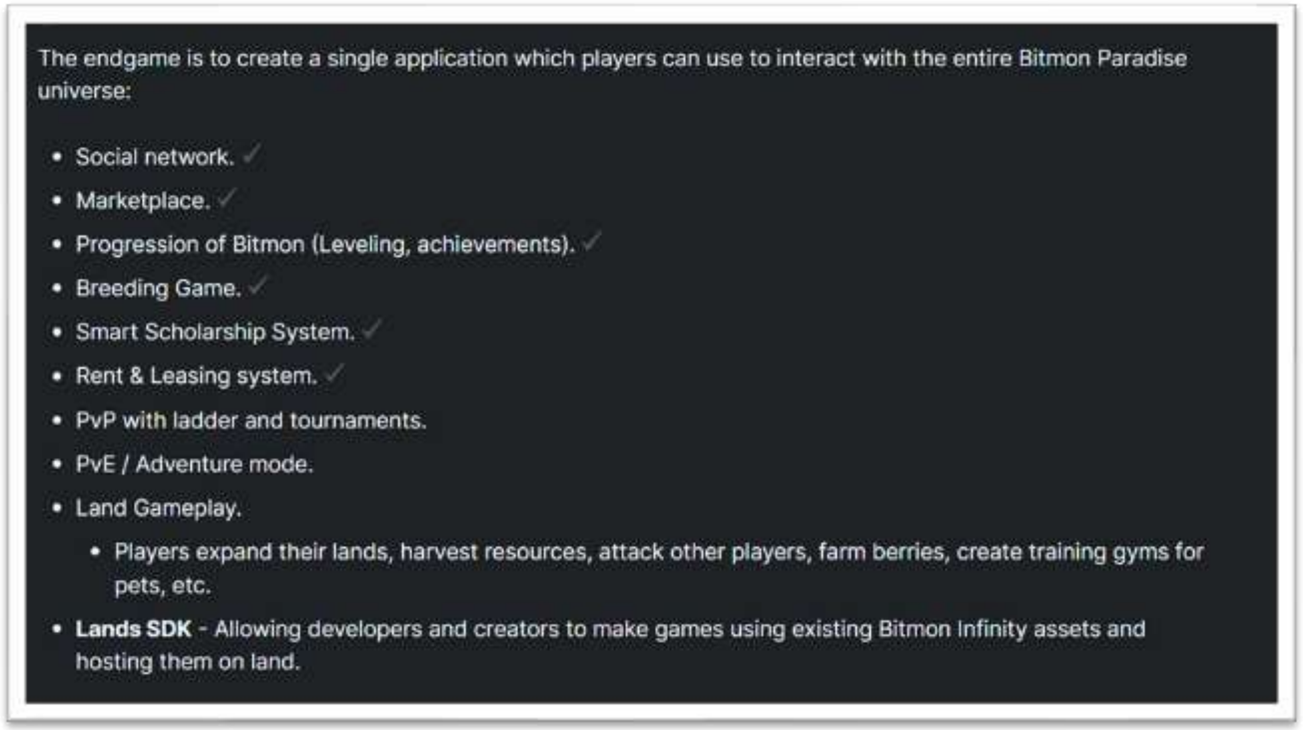
# Roadmap

*A roadmap was found on the official website, we have conveniently placed it on this page for your viewing.*



## Roadmap

- July 2021: Concept and Idea - Development begins. ✓
- Q1 2022: Presentation of token, and Bitmon. ✓
- May 2022: Smart Contracts Audits. ✓
- June 2022: alpha of the game "GARDEN MODE". ✓
  - Andorid (Google Play). ✓
  - WebGL (Mac, Windows & Linux). ✓
- September 2022: My Account release. ✓
- Q4 2022: In-House Bitmon-NFT Marketplace released, breeding game release. ✓
- April 2023: Idle battle game release.
- Q3 2023: Presale begins. Smart Pro Schoolar Dashboard realese.
- July 2023: Bitmon Clash release.
- July 2023: Smart Pro Schoolar Dashboard realese..
- December 2023: Gravit.y HUB Alpha release
- Q1 2024: Alpha Adventure Mode release.
- Q2 2024: Presentation of the Lands.
- Q3 2024: $GVT staking & ecosystem begins.
  - Governance.
  - Play to Earn.
  - Mainstream release of Bitmon Paradise on iOS/Android.
- Q4 2024: Land gameplay.
- Q1 2025: Lands SDK Alpha.



The endgame is to create a single application which players can use to interact with the entire Bitmon Paradise universe:

- Social network. ✓
- Marketplace. ✓
- Progression of Bitmon (Leveling, achievements). ✓
- Breeding Game. ✓
- Smart Scholarship System. ✓
- Rent & Leasing system. ✓
- PvP with ladder and tournaments.
- PvE / Adventure mode.
- Land Gameplay.
  - Players expand their lands, harvest resources, attack other players, farm berries, create training gyms for pets, etc.
- **Lands SDK** - Allowing developers and creators to make games using existing Bitmon Infinity assets and hosting them on land.

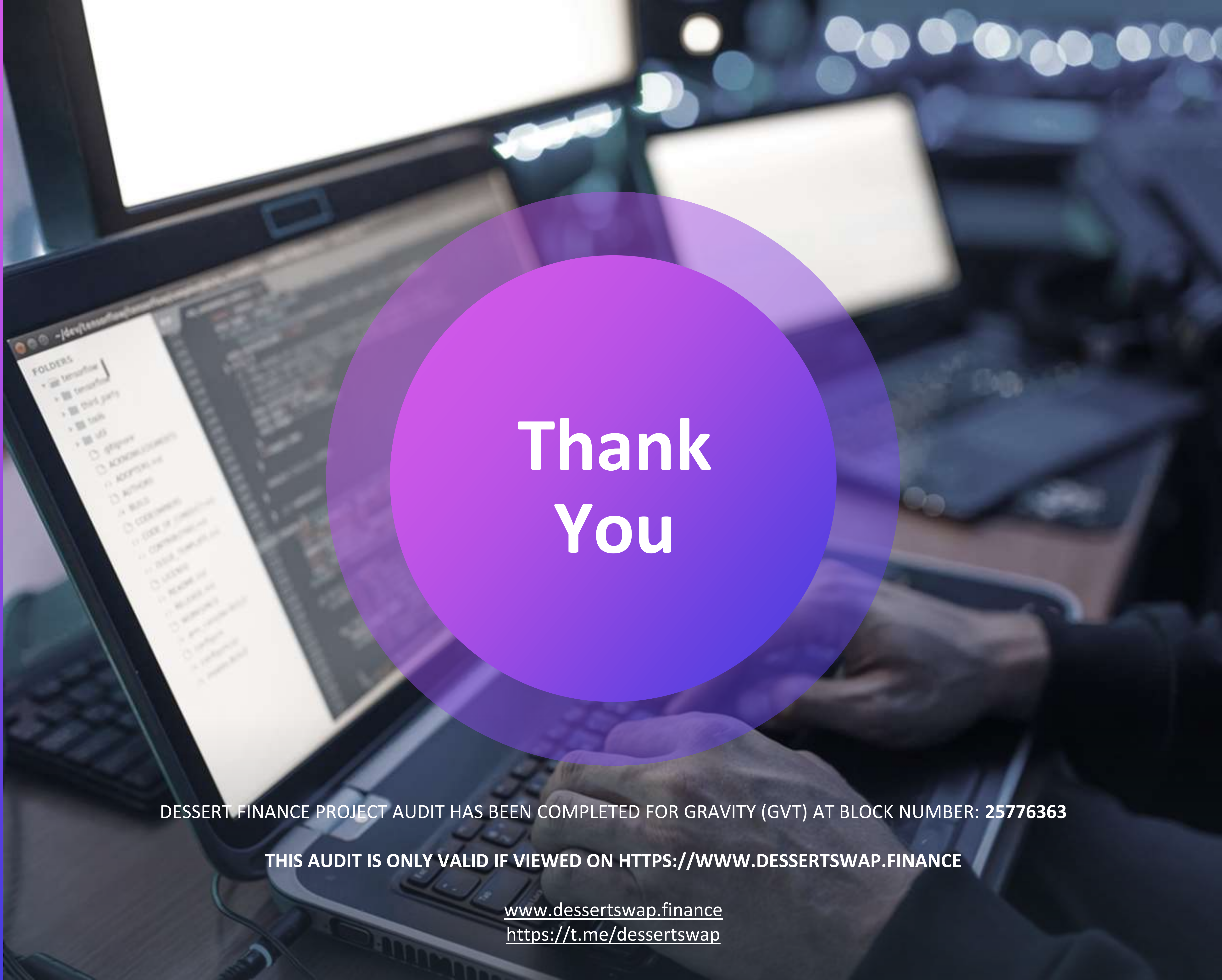
# Disclaimer

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 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 GRAVITY (GVT) AT BLOCK NUMBER: **25776363**

**THIS AUDIT IS ONLY VALID IF VIEWED ON HTTPS://WWW.DESSERTSWAP.FINANCE**

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