



ETHOS  
AUDIT

# Baked Beans

Audit Report

Jan. 6, 2022

# Contents

Executive Summary .....	3
Audit Details .....	3
Methodology .....	3
Contract Details .....	4
Result Summary .....	4
Risk Levels .....	4
Issues Reported .....	5
Issues Summary .....	5
Detailed Findings.....	5
BB-0 – Variables can be declared as ‘constant’ .....	5
BB-1 – Missing Event emissions .....	5
BB-2 – Function initializing state .....	6
BB-3 – Functions that could be declared external.....	6
Code Documentation .....	6
Adherence to Specifications.....	6
Adherence to Best Practices.....	7
Appendix .....	7
Revision History .....	7
Functions.....	7
Global Variables .....	7

# Executive Summary

## Audit Details

Project Name	Dextera
Codebase	<a href="https://www.bscscan.com/address/0xe2d26507981a4daaaa8040bae1846c14e0fb56bf#code">https://www.bscscan.com/address/0xe2d26507981a4daaaa8040bae1846c14e0fb56bf#code</a>
Source Code	BakedBeans.sol
Initial Audit Date	January 6, 2022
Revision Dates	-
Methodology	Manual

## Methodology

This audit's objectives are to evaluate:

- Security-related issues
- Code quality
- Relevant documentation
- Adherence to specifications
- Adherence to best practices

This audit examines the possibility of issues existing along the following vectors (but not limited to):

- Single & Cross-Function Reentrancy
- Front Running (Transaction Order Dependence)
- Timestamp dependence
- Integer Overflow and Underflow
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Number rounding errors
- DoS with (Unexpected) Revert
- DoS with Block Gas Limit
- Insufficient gas grieving
- Forcibly sending native currency
- Logical oversights
- Access control
- Centralization of power
- Logic-Specification Contradiction
- Functionality duplication
- Malicious token minting

The code review conducted for this audit follows the following structure:

1. Review of specifications, documentation to assess smart contract functionality
2. Manual, line-by-line review of code
3. Code's adherence to functionality as presented by documentation
4. Automated tool-driven review of smart contract functionality
5. Assess adherence to best practices
6. Provide actionable recommendations

## Contract Details

Contract ID	0xe2d26507981a4daaaa8040bae1846c14e0fb56bf
Network	BSC
Language	Solidity
Compiler	v0.8.9+commit.e5eed63a
Verification Date	Nov. 6, 2021
Contract Type	Utility Contract
Libraries	Custom

## Result Summary

Ethos' audit of the Baked Beans smart contract has concluded with a **POSITIVE** result. The initial review identified a number of non-critical issues. The remaining report includes all issues identified in the initial review, as well as the revised status post resolution by the team.

## Risk Levels



### LOW

The issue is informational and does not pose an immediate risk, but is relevant to security best practices.



### MEDIUM

The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low impact in view of the client's business circumstances.



### HIGH

The issue puts a subset of users' sensitive functionality at risk, would be detrimental for the client's reputation if exploited, or is reasonably likely to lead to moderate financial impact.



### EXTREME

The issue puts a large number of users' sensitive functionality at risk, or is reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.

## Issues Reported

Severity	Unresolved	Acknowledged	Resolved
Extreme	0	0	0
High	0	0	0
Medium	0	0	0
Low	0	4	0

## Issues Summary

ID	Title	Severity	Status
BB-0	Variables can be declared as 'constant'	Low	Resolved
BB-1	Missing Event emissions	Low	Acknowledged
BB-2	Function initializing state	Low	Acknowledged
BB-3	Functions that could be declared external	Low	Acknowledged

## Detailed Findings

### BB-0 – Variables can be declared as 'constant'

**Severity:** Low

**Status:** Acknowledged

**Description:** Several global variables that are initialized with a value on definition can be declared as constant since they are never changed throughout the body of the smart contract.

**Risk:** This is a minor gas optimization issue.

**Recommendation:** We recommend declaring these variables as 'constant' if they aren't going to be changed.

### BB-1 – Missing Event emissions

**Severity:** Low

**Status:** Acknowledged

**Description:** There are several functions that change state variables, however, they do not emit events to pass the changes out of chain.

**Risk:** Not emitting an event from functions that impose changes to state variables could result in a lack of functionality often required for sound logic and functionality within external applications calling on smart contract functions.

**Recommendation:** We recommend emitting events for all essential state variables that are possible to be changed during runtime.

## BB-2 – Function initializing state

**Severity:** Low

**Status:** Acknowledged

**Description:** A state variable is initialized through function calls that are not pure/constant, or that use non-constant state variables

**Risk:** Users might intend a function to return a value a state variable can initialize with, without realizing the context for the contract is not fully initialized.

**Recommendation:** Remove any initialization of state variables via non-constant state variables or function calls. If variables must be set upon contract deployment, locate initialization in the constructor instead.

## BB-3 – Functions that could be declared external

**Severity:** Low

**Status:** Acknowledged

**Description:** Several functions are declared as public visibility, however, since they are never called by the contract they should be declared external.

**Risk:** This is a gas optimization issue.

**Recommendation:** We recommend that functions that are never called by the contract to be declared as external to save gas.

## Code Documentation

The code has a moderate amount of comments. This could be improved in order to help others understand the contract.

## Adherence to Specifications

The smart contract adheres to the smart contract functionality described by the Baked Beans team and is in line with its intended usage.

## Adherence to Best Practices

The smart contract adheres to the majority of best practices associated with a standard BEP-20 token. The few things that don't follow the best practices are noted in the automated review.

## Appendix

### Revision History

Initial Review: Jan. 6, 2021 – Audit results delivered for contract ID: 0xe2d26507981a4daaaa8040bae1846c14e0fb56bf

### Functions

Function	Parameters	Visibility	Modifiers	Returns	Requires	Events
constructor						
hatchEggs	address ref	public				initialized
sellEggs		public				initialized
beanRewards	address ref	public		uint256		
buyEggs	address ref	public	payable			initialized
calculateTrade	uint256 rt, uint256 rs, uint256 bs	private		uint256		
calculateEggSell	uint256 eggs	public		uint256		
calculateEggBuy	uint256 eth, uint256 contractBalance	public		uint256		
calculateEggBuySimple	uint256 eth	public		uint256		
devFee	uint256 amount	private		uint256		
seedMarket		public	payable onlyOwner			marketEggs == 0
getBalance		public		uint256		
getMyMiners	address adr	public		uint256		
getMyEggs	address adr	public		uint256		
getEggsSinceLastHatch	address adr	public		uint256		
min	uint256 a, uint256 b	private		uint256		

### Global Variables

Variable	Type	Visibility	Read by Functions	Written by Functions
EGGS_TO_HATCH_1MINERS	uint256	private		
PSN	uint256	private		
PSNH	uint256	private		
devFeeVal	uint256	private		
initialized	bool	private		
recAdd	address payable	private		

hatcheryMiners	mapping (address => uint256)	private
claimedEggs	mapping (address => uint256)	private
lastHatch	mapping (address => uint256)	private
referrals	mapping (address => address)	private
marketEggs	uint256	private