



# CFG NINJA AUDITS

Security Assessment

## JayPowell AI Token

October 11, 2023

Audit Status: Pass

Audit Edition: Pinksale



POWERED BY  
**BLADE POOL**

# Risk Analysis

## Classifications of Manual Risk Results

Classification	Description
🔴 Critical	Danger or Potential Problems.
🟠 High	Be Careful or Fail test.
🟢 Low	Pass, Not-Detected or Safe Item.
ℹ️ Informational	Function Detected

## Manual Code Review Risk Results

Contract Priviledge	Description
🟢 Buy Tax	4%
🟢 Sale Tax	4%
🟢 Cannot Sale	Pass
🟢 Cannot Sale	Pass
🟢 Max Tax	25%
ℹ️ Modify Tax	Yes
🟢 Fee Check	Pass
🟢 Is Honeypot?	Not Detected
🟢 Trading Cooldown	Not Detected
🟢 Can Pause Trade?	Not Detected.



Contract Priviledge	Description
● Pause Transfer?	Not Detected
● Max Tx?	Pass
● Is Anti Whale?	Not Detected
● Is Anti Bot?	Detected
● Is Blacklist?	Not Detected
● Blacklist Check	Pass
● Is Whitelist?	Not Detected
● Can Mint?	Pass
● Is Proxy?	Not Detected
● Can Take Ownership?	Not Detected
● Hidden Owner?	Not Detected
● Owner	0x65AACCB4fDF8281BeA71830562651fA8c35DB2CF
● Self Destruct?	Not Detected
● External Call?	Detected
● Other?	Detected
● Holders	1
● Auditor Confidence	Low Risk

The following quick summary it's added to the project overview; however, there are more details about the audit and its results. Please read every detail.



# Project Overview

## Token Summary

Parameter	Result
Address	0x64B1ACDFd7e147d52C9141C07671af7303e2f0ab
Name	JayPowell AI
Token Tracker	JayPowell AI (JPWL)
Decimals	18
Supply	42,000,000,000
Platform	Binance Smart Chain
compiler	v0.8.4+commit.c7e474f2
Contract Name	AntiBotBABYTOKEN
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	<a href="https://bscscan.com/token/0x64B1ACDFd7e147d52C9141C07671af7303e2f0ab#code">https://bscscan.com/token/0x64B1ACDFd7e147d52C9141C07671af7303e2f0ab#code</a>
Payment Tx	Corporate





## Main Contract Assessed

### Contract Name

Name	Contract	Live
JayPowell AI	0x64B1ACDFd7e147d52C9141C07671af7303e2f0ab	Yes

## TestNet Contract was Not Assessed

### Solidity Code Provided

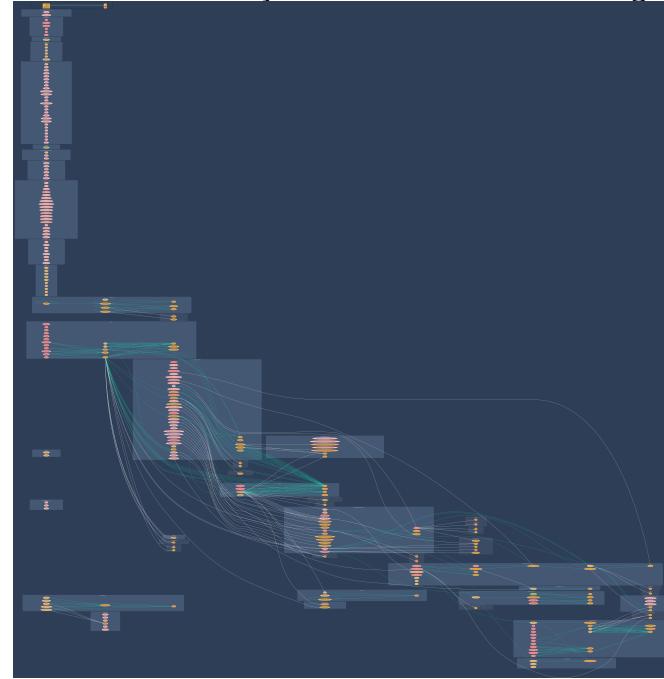
SollID	File Sha-1	FileName
ABBT	ec0e5896c54f0046e5517ed944969d349292340	AntibotBabyToken.sol
ABBT		





# Call Graph

The contract for JayPowell AI has the following call graph structure.



# Smart Contract Vulnerability Checks

The Smart Contract Weakness Classification Registry (SWC Registry) is an implementation of the weakness classification scheme proposed in EIP-1470. It is loosely aligned to the terminologies and structure used in the Common Weakness Enumeration (CWE) while overlaying a wide range of weakness variants that are specific to smart contracts.

ID	Severity	Name	File	location
SWC-100	Pass	Function Default Visibility	AntibotBabyToken.sol	L: 0 C: 0
SWC-101	Pass	Integer Overflow and Underflow.	AntibotBabyToken.sol	L: 0 C: 0
SWC-102	Pass	Outdated Compiler Version file.	AntibotBabyToken.sol	L: 0 C: 0
SWC-103	Pass	A floating pragma is set.	AntibotBabyToken.sol	L: 0 C: 0
SWC-104	Pass	Unchecked Call Return Value.	AntibotBabyToken.sol	L: 0 C: 0
SWC-105	Pass	Unprotected Ether Withdrawal.	AntibotBabyToken.sol	L: 0 C: 0
SWC-106	Pass	Unprotected SELFDESTRUCT Instruction	AntibotBabyToken.sol	L: 0 C: 0
SWC-107	Pass	Read of persistent state following external call.	AntibotBabyToken.sol	L: 0 C: 0
SWC-108	Pass	State variable visibility is not set..	AntibotBabyToken.sol	L: 0 C: 0
SWC-109	Pass	Uninitialized Storage Pointer.	AntibotBabyToken.sol	L: 0 C: 0
SWC-110	Pass	Assert Violation.	AntibotBabyToken.sol	L: 0 C: 0



ID	Severity	Name	File	location
SWC-111	Pass	Use of Deprecated Solidity Functions.	AntibotBabyToken.sol	L: 0 C: 0
SWC-112	Pass	Delegate Call to Untrusted Callee.	AntibotBabyToken.sol	L: 0 C: 0
SWC-113	Pass	Multiple calls are executed in the same transaction.	AntibotBabyToken.sol	L: 0 C: 0
SWC-114	Pass	Transaction Order Dependence.	AntibotBabyToken.sol	L: 0 C: 0
SWC-115	Pass	Authorization through tx.origin.	AntibotBabyToken.sol	L: 0 C: 0
SWC-116	Pass	A control flow decision is made based on The block.timestamp environment variable.	AntibotBabyToken.sol	L: 0 C: 0
SWC-117	Pass	Signature Malleability.	AntibotBabyToken.sol	L: 0 C: 0
SWC-118	Pass	Incorrect Constructor Name.	AntibotBabyToken.sol	L: 0 C: 0
SWC-119	Pass	Shadowing State Variables.	AntibotBabyToken.sol	L: 0 C: 0
SWC-120	Pass	Potential use of block.number as source of randomness.	AntibotBabyToken.sol	L: 0 C: 0
SWC-121	Pass	Missing Protection against Signature Replay Attacks.	AntibotBabyToken.sol	L: 0 C: 0
SWC-122	Pass	Lack of Proper Signature Verification.	AntibotBabyToken.sol	L: 0 C: 0
SWC-123	Pass	Requirement Violation.	AntibotBabyToken.sol	L: 0 C: 0
SWC-124	Pass	Write to Arbitrary Storage Location.	AntibotBabyToken.sol	L: 0 C: 0



ID	Severity	Name	File	location
SWC-125	Pass	Incorrect Inheritance Order.	AntibotBabyToken.sol	L: 0 C: 0
SWC-126	Pass	Insufficient Gas Griefing.	AntibotBabyToken.sol	L: 0 C: 0
SWC-127	Pass	Arbitrary Jump with Function Type Variable.	AntibotBabyToken.sol	L: 0 C: 0
SWC-128	Pass	DoS With Block Gas Limit.	AntibotBabyToken.sol	L: 0 C: 0
SWC-129	Pass	Typographical Error.	AntibotBabyToken.sol	L: 0 C: 0
SWC-130	Pass	Right-To-Left-Override control character (U+202E).	AntibotBabyToken.sol	L: 0 C: 0
SWC-131	Pass	Presence of unused variables.	AntibotBabyToken.sol	L: 0 C: 0
SWC-132	Pass	Unexpected Ether balance.	AntibotBabyToken.sol	L: 0 C: 0
SWC-133	Pass	Hash Collisions with Multiple Variable Length Arguments.	AntibotBabyToken.sol	L: 0 C: 0
SWC-134	Pass	Message call with hardcoded gas amount.	AntibotBabyToken.sol	L: 0 C: 0
SWC-135	Pass	Code With No Effects (Irrelevant/Dead Code).	AntibotBabyToken.sol	L: 0 C: 0
SWC-136	Pass	Unencrypted Private Data On-Chain.	AntibotBabyToken.sol	L: 0 C: 0

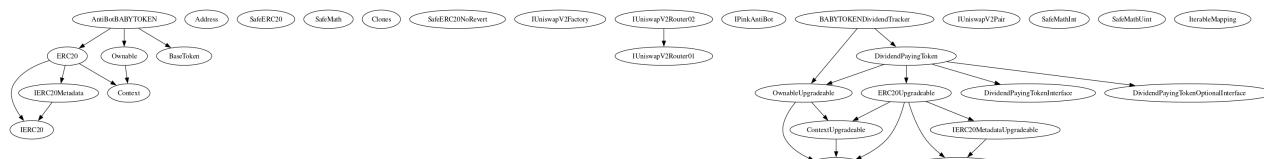
We scan the contract for additional security issues using MYTHX and industry-standard security scanning tools.





# Inheritance

The contract for Jay Powell AI has the following inheritance structure.



# Smart Contract Advance Checks

ID	Severity	Name	Result	Status
JPWL-01	Low	Potential Sandwich Attacks.	Pass	Not Detected
JPWL-02	Informational	Function Visibility Optimization	Pass	Not Detected
JPWL-03	Low	Lack of Input Validation.	Pass	Not Detected
JPWL-04	High	Centralized Risk In addLiquidity.	Pass	Not Detected
JPWL-05	Low	Missing Event Emission.	Pass	Not Detected
JPWL-06	Low	Conformance with Solidity Naming Conventions.	Pass	Not Detected
JPWL-07	Low	State Variables could be Declared Constant.	Pass	Not Detected
JPWL-08	Low	Dead Code Elimination.	Pass	Not Detected
JPWL-09	High	Third Party Dependencies.	Pass	Not Detected
JPWL-10	High	Initial Token Distribution.	Pass	Not Detected
JPWL-11	High	claimStuckTokens can claim own tokens.	Pass	Not Detected
JPWL-12	High	Centralization Risks In The X Role	Pass	Not Detected
JPWL-13	Informational	Extra Gas Cost For User..	Fail	Detected
JPWL-14	Medium	Unnecessary Use Of SafeMath	Fail	Detected
JPWL-15	Medium	Symbol Length Limitation due to Solidity Naming Standards.	Pass	Not Detected



ID	Severity	Name	Result	Status
JPWL-16	Medium	Taxes can be up to 100%	Pass	Not Detected
JPWL-17	Logical Issue	Highly Permissive Role Access.;	Pass	Not Detected
JPWL-18	Critical	Stop Transactions by using Enable Trade.	Pass	Not Detected



## JPWL-13 | Extra Gas Cost For User.

Category	Severity	Location	Status
Logical Issue	<span>i</span> Informational	AntibotBabyToken.sol: L: 3326, C: 0	<span>🕒</span> Detected

### Description

The user may trigger a tax distribution during the transfer process, which will cost a lot of gas and it is unfair to let a single user bear it.

### Remediation

We advise the client to make the owner responsible for the gas costs of the tax distribution.

### Project Action



## JPWL-14 | Unnecessary Use Of SafeMath

Category	Severity	Location	Status
Logical Issue	<span style="color: yellow;">●</span> Medium	AntibotBabyToken.sol: L: 913 C: 12	<span style="color: green;">🔗</span> Detected

### Description

The SafeMath library is used unnecessarily. With Solidity compiler versions 0.8.0 or newer, arithmetic operations

will automatically revert in case of integer overflow or underflow.

```
library SafeMath {
```

An implementation of SafeMath library is found.

```
using SafeMath for uint256;
```

SafeMath library is used for uint256 type in contract.

### Remediation

We advise removing the usage of SafeMath library and using the built-in arithmetic operations provided by the

Solidity programming language

### Project Action



# Technical Findings Summary

## Classification of Risk

Severity	Description
🔴 Critical	Risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
🟠 High	Risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
🟡 Medium	Risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform
🟢 Low	Risks can be any of the above but on a smaller scale. They generally do not compromise the overall integrity of the Project, but they may be less efficient than other solutions.
ℹ️ Informational	Errors are often recommended to improve the code's style or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

## Findings

Severity	Found	Pending	Resolved
🔴 Critical	0	0	0
🟠 High	0	0	0
🟡 Medium	1	0	0
🟢 Low	0	0	0
ℹ️ Informational	1	0	0
Total	2	0	0



# Social Media Checks

Social Media	URL	Result
Twitter	<a href="https://twitter.com/JPWLA1">https://twitter.com/JPWLA1</a>	Pass
Other	<a href="https://www.youtube.com/@JayPowellAI">https://www.youtube.com/@JayPowellAI</a>	Pass
Website	<a href="https://jpowellai.com/">https://jpowellai.com/</a>	Pass
Telegram	<a href="https://t.me/JayPowellAI_Official">https://t.me/JayPowellAI_Official</a>	Pass

We recommend to have 3 or more social media sources including a completed working websites.

**Social Media Information Notes:**

**Auditor Notes:** undefined

**Project Owner Notes:**



# Assessment Results

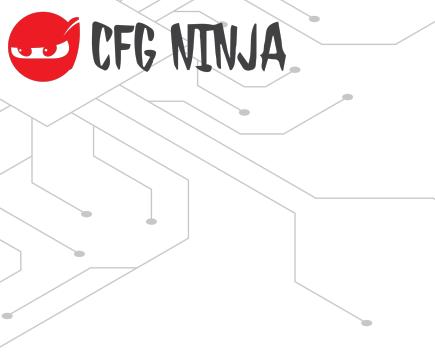
## Score Results

Review	Score
Overall Score	92/100
Auditor Score	80/100
Review by Section	Score
Manual Scan Score	24/33
SWC Scan Score	37 /37
Advance Check Score	31 /30

The Following Score System Has been Added to this page to help understand the value of the audit, the maximum score is 100, however to attain that value the project must pass and provide all the data needed for the assessment. Our Passing Score has been changed to 80 Points, if a project does not attain 80% is an automatic failure. Read our notes and final assessment below.

## Audit Passed





## Assessment Results

### Important Notes:

- No issues or vulnerabilities were found.
- This is a Pinksale Generated BabyToken.
- Please DYOR on the project.
- The contract gives Ethereum <https://bscscan.com/address/0x2170ed0880ac9a755fd29b2688956bd959f933f8>
- this contract depends on volume and buy/sale then distribution of rewards may happen.
- This type of contract may fail if the fees are set to 0.

**Auditor Score =80**  
**Audit Passed**



# Appendix

## Finding Categories

### Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

### Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

### Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how `block.timestamp` works.

### Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being invokeable by anyone under certain circumstances.

### Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

### Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

### Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code, such as a constructor assignment imposing different require statements on the input variables than a setter function.



## Coding Best Practices

ERC 20 Coding Standards are a set of rules that each developer should follow to ensure the code meet a set of criterias and is readable by all the developers.





## Disclaimer

CFGNINJA has conducted an independent security assessment to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the reviewed code for the scope of this assessment. This report does not constitute agreement, acceptance, or advocacy for the Project, and users relying on this report should not consider this as having any merit for financial advice in any shape, form, or nature. The contracts audited do not account for any economic developments that the Project in question may pursue, and the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude, and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are entirely free of exploits, bugs, vulnerabilities or depreciation of technologies.

All information provided in this report does not constitute financial or investment advice, nor should it be used to signal that any persons reading this report should invest their funds without sufficient individual due diligence, regardless of the findings presented. Information is provided 'as is', and CFGNINJA is under no covenant to audited completeness, accuracy, or solidity of the contracts. In no event will CFGNINJA or its partners, employees, agents, or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions or actions with regards to the information provided in this audit report.

The assessment services provided by CFGNINJA are subject to dependencies and are under continuing development. You agree that your access or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies with high levels of technical risk and uncertainty. The assessment reports could include false positives, negatives, and unpredictable results. The services may access, and depend upon, multiple layers of third parties.

