# Preliminary Notes: MoonVault

## Brewlabs Services Hub

Summary: MoonVault is a BSC farming project that allows lenders earn rewards from depositing their assets into farming vaults. The yield generated by the vault is determined by the vault strategy. It is written in Solidity and compiled using build 0.8.0.

*Audit package:* Standard
*Deployment Status:* Not Deployed
*Files Audited:* VaultStrategyV1.sol; VaultV1.sol. All imported files (excluding OpenZeppelin) for these two contracts were also audited.
*Audit Result:* **PASSED**

***Disclaimer:*** *This audit documentation is for discussion purposes only. The scope of this audit was to analyze and document MoonVault smart contract codebase for quality, security, and accuracy. This audit guarantees that your code has been revised by an expert.*

**Overview of Audit**
No major or medium security issues were identified.
Good use of comments, input validation and events.

**Security issues**
Low level security issued detected:

1. Floating pragma set (^*{solidity compiler version}. If relying on byte-code level verification, best practice is to lock the compiler version prior to deployment.
2. SafeApprove usage is discouraged due to possible unfortunate transaction ordering. Dev Team have already integrated mitigation recommendation by setting the allowance to zero before resetting the allowance to the preferred value.
3. Centralisation risk for inCaseTokensGetStuck() and panic() where Admin and SecurityMod roles can impact the contract state by pausing and withdrawing tokens in a vault. While the purpose of these methods are understood, they do present a risk to lenders. To mitigate risk, ensure the Dev Team is doxxed or consider (Know Your Customer) KYC services.
4. The VaultStrategy contract interacts with the third-party uniswap protocol. Third-party protocols can be compromised, as well as subject to upgrades which can possibly result in possibly severe impacts, such as increasing fees, LP pool migration, lost or stolen assets etc. Dev Team should monitor any updates or changes made to third party integrations to mitigate risk.
5. No access restriction for swap functions (lines 488 - 623), and functions can be called from derived contracts. Consider adding in access restriction.

**Logical Issues:**

1. Dev team to confirm if vault withdraws should available when the strategies are paused and the balanceVaultBefore >= withdrawAmount

**Continued,**

**Code Style:**

1. Revise contract for spelling mistakes, eg. setCompundRewardNative, _addresss are misspelt.
2. Use inclusive terminology that does not promote negative associations with the colours 'black' or 'white' eg. requireWhitelistedContract

**Conclusion:** The Brewlabs team thank you for the opportunity to review and audit your smart contact code. The data from this report will be formalised in the audit publication for your community. Keep in touch as we offer discounts for repeat business and on a range of other services!

Maverick & Peach
The Brewlabs Team.