LIFESTONE LABS

FundMe.Cash

LIFESTONE LABS

Project Audit Report

CashStarter

Audit Date: November 3, 2025

Auditor: Kyle Wildeman

Table of Contents

Report Sections

Audit Report Disclaimer

Purpose of This Audit

This is a self-audit by the creator to use as an example of the types of audit reports that Lifestone Labs provides. The purpose is to identify potential vulnerabilities, logic errors, and security concerns that may have been overlooked during internal development and testing.

Scope of Analysis

This audit examines the smart contract code for technical vulnerabilities, logic flaws, and potential attack vectors at the contract level. It does not include any analysis of frontend or backend infrastructure. Additionally, it attempts to highlight any trust assumptions or centralization risks that users should be aware of when interacting with the contracts.

Methodology

Manual code review is performed to identify security vulnerabilities, logic errors, and potential attack vectors. This process includes:

- Manual analysis of smart contract code
- Diagramming the projects smart contracts, interactions, and transactions
- Identification of common smart contract vulnerabilities
- Recommendations for security improvements

Limitations and Disclaimer

This audit does not guarantee the security of the audited contracts. Security audits are inherently limited by time, scope, and the possibility of undiscovered vulnerabilities. The absence of identified issues does not imply the absence of vulnerabilities. Users should conduct their own due diligence and consider additional security measures when interacting with smart contracts.

CashStarter Audit Report

Project Information

Project Name:

CashStarter

Audit Date:

November 3, 2025

Auditor:

Kyle Wildeman

Language:

CashScript 0.10.0

Project Symbol:

None

Project Website:

https://fundme.cash

Project Logo:

Code Snapshot:

5324100a08e37f5bf0e8bb542baa3230581e8a87

Project Description

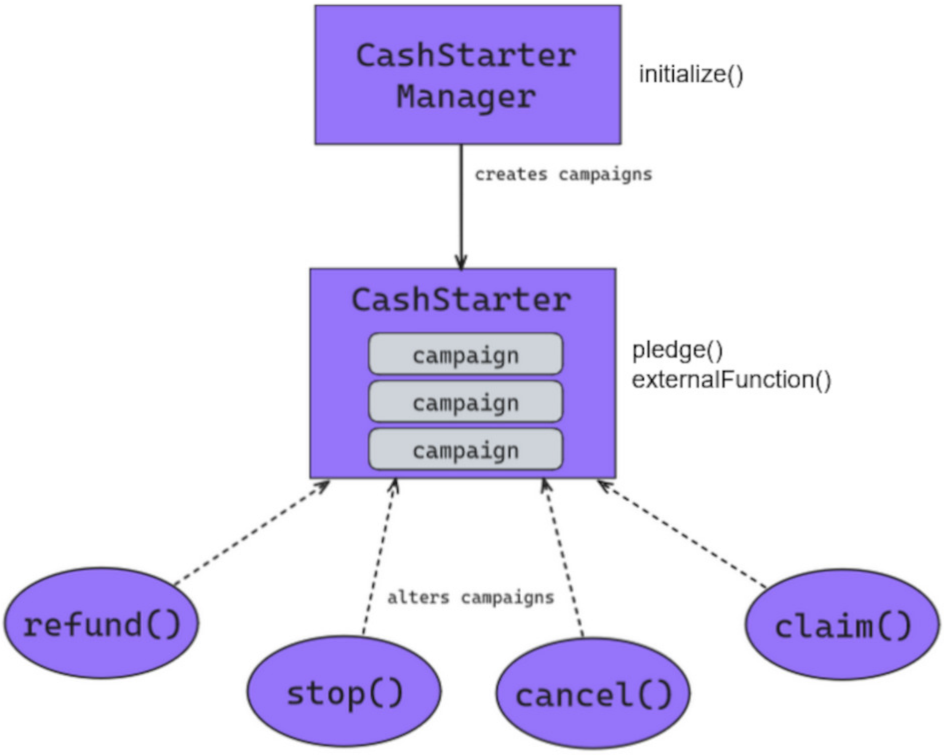
CashStarter is an on-chain, non-custodial crowdfunding app comprised of 6 contracts on Bitcoin Cash. Users create a campaign with a BCH goal amount and if it is reached or exceeded the creator can claim the funds. Users who pledge receive a NFT that lets them refund their pledge at any time before the campaign is claimed. FundMe.Cash is a frontend that can be used to interact with the contracts.

Findings and Severity

HIGH: 0 MEDIUM: 1 LOW: 1 INFO: 0

Transactions and Trust Levels

OPERATOR TRUSTED: 0 FRONTEND TRUSTED: 3 TEMPLATE TRUSTED: 0 TRUSTLESS: 3


CashStarter Manager
initialize()
creates campaigns
CashStarter
campaign
campaign
campaign
pledge()
externalFunction()
alters campaigns
refund()
stop()
cancel()
claim()

Scope Details

This section lists contracts and their functions covered in this audit. Function descriptions are provided by the project.

CashStarter p03u4v845j4rhzyc63cgm0arksfx5u3atkvz43xzdy0r8pql4qm37j50tje8p

pledge

User pledges an amount of BCH to a campaignNFT on the CashStarter contract. The BCH is added to the campaignNFT and the user is given a receiptNFT.

externalFunction

Interact with another contract of the CashStarter category with Minting capability (limited to the CashStarter, cancel, claim, refund, and stop contracts)

CashStarterManager p0mhegq4agfajfjda5al2nwcz6v5ztvct7p3tdp4kfzpqmpv8v9xu5j8aunkq

initialize

Creates a new campaign UTXO on the CashStarter contract. User provides a UTXO to cover fees and ensures their address is the one that will receive the raised funds. Parameters are used to provide the campaigns ending block and funding amount. Built-in service provider fee (0 to 0.01 BCH) to incentivize website frontends.

CashStarterCancel pdwugz8d34vclx45w5negqp6rx3535z0k9uynlp7ey4p6g6lyantgfqcj7n2n

cancel

Creator removes minting capability from an active campaignNFT. Prevents pledge(), fail(), cancel() from being called on the campaign. Still allows refund() and claim() to be called on the campaign.

CashStarterClaim pw2gwuvr5s5uz4g8qylu44ut6dkp26s56w9wvpa3dl95r9nj47s7q6evhtleg

claim

Creator claims campaign. Campaign must be past the fundTarget. ServiceProvider receives a fee (max 0.01 BCH)

CashStarterRefund pvpf5ppmx2dw60u5qm2jcarm0df6f7ldzqk8kr25q4gzmm4802v7wmfcv4eyz

refund

User refunds their pledge from a campaignNFT. If it's the last pledge on a campaign that has been stop()'d or cancel()'d then the campaign gets burned.

CashStarterStop p0z4d9acwvhzyr5qqtqt7ejj3ur4na3j6u2lpeg0rujz5ysjay86qgea803ql

stop

Removes the minting capability from a campaignNFT. Prevents pledge(), cancel(), and stop(). Still allows refund() and claim().

Transactions

This section details the dapps transactions and their level of required trust.

> **Note: All transactions made by the user still need to be Approved by the user unless they have enabled an auto-Approve feature in their wallet. This means that users do have the opportunity to verify whether a transaction is incorrect or malicious before they approve it. However, we rank the Trust Classification of transactions under the worst-case assumption, which is a user that does not manually verify every transaction request themselves before approving it.**

Trust Levels:

Level	Description
OPERATOR TRUSTED	The user must trust the actions of another party (team, admins, external service) to behave correctly. This may include a systemic risk of loss or harm if the party behaves incorrectly.
FRONTEND TRUSTED	The user must trust the frontend interface to build some parts of the transaction correctly since the contracts do not enforce those settings.
TEMPLATE TRUSTED	The user must trust the templates are correct and not maliciously designed.
TRUSTLESS	The contracts enforce all critical transaction building decisions so that no user loss or harm can occur, regardless of the frontend or wallet behaviour.

1. Create Campaign

FRONTEND TRUSTED

User creates a new campaign on the CashStarter contract.

Functions
CashStarterManager.initialize()

Trust Level Explanation:

> The frontend is being trusted to enter the correct amount of BCH the campaign is going to raise and what block the campaign will end. Besides these, the transaction is Trustless.

2. Pledge to Campaign

FRONTEND TRUSTED

A user pledges an amount of their BCH to an active campaign.

Functions
CashStarter.pledge()

Trust Level Explanation:

> The frontend is being trusted to create the correct amount of change BCH.

3. Cancel Campaign

TRUSTLESS

Campaign creator stops a currently active campaign

Functions

CashStarter.externalFunction() CashStarterCancel.cancel()

Trust Level Explanation:

> All transaction details are enforced by the contract.

4. Stop a Campaign

TRUSTLESS

Removes the minting capability from a campaign that failed to reach its goal in time. Prevents pledge(), cancel(), and stop(). Still allows refund() and claim().

Functions

CashStarter.externalFunction() CashStarterStop.stop()

Trust Level Explanation:

> All transaction details are enforced by the contract.

5. Refund a Pledge

TRUSTLESS

User refunds their pledge from a campaign.

Functions

CashStarter.externalFunction() CashStarterRefund.refund()

Trust Level Explanation:

> All transaction details are enforced by the contract.

6. Claim a Campaign

FRONTEND TRUSTED

Campaign creator claims a campaign that has successfully raised the targeted amount of BCH.

Functions

CashStarter.externalFunction() CashStarterClaim.claim()

Trust Level Explanation:

> The address of the service provider and their fee amount is set by the transaction builder (so ... the service provider). At worst, the service providers frontend could be compromised and have their fee stolen and set to the maximum (5%) when the user was expecting a lower percentage. Besides this, the transaction would be considered Trustless.

Findings

This section lists the potential issues found during the audit.

Severity	Description
HIGH	Findings that can result in significant loss of user funds or cause unintended contract manipulation which undermines the integrity of the project.
MEDIUM	Findings that can result in loss of user funds in a more limited scope or allows contract manipulation which can cause unintended consequences, but doesn't undermine the integrity of the entire project.
LOW	Minor, non-critical findings that allow or cause unintended consequences, such as an edge-case that prevents a user from performing a certain expected transaction, but can be worked around with no harm or loss.
INFO	Minor findings which are unlikely to have any negative impact. Observations of the design's limitations or restrictions.

1. Pledging doesn't enforce BCH change amount

MEDIUM

If a user pledges a portion of their UTXO rather than the full amount they would expect some BCH returned as change. The contracts enforce that if there is a change amount it must be sent back to the users address, but it doesn't enforce the amount.

Impact:

The frontend could set an incorrect change amount, or no change amount. Any BCH not returned to the user would be lost, automatically added to the transaction fee.

Recommendation:

> Enforce that a majority of the expected change must go back to the users address, e.g.
>
> ```
> require(tx.outputs[2].value >= tx.inputs[1] - pledgeAmount - 2000);
> ```

Status: Acknowledged

Project Response:

Verified the current FundMe.Cash website does calculate user change amount correctly. Will enforce user change on a future version of the contracts.

2. New Campaigns & Frontend Trust

LOW

When users create new campaigns they are trusting the frontend to provide the users intended values for the amount of BCH the campaign is attempting to raise, and the block the campaign will end at.

Impact:

At worst, the frontend could create a campaign that will never be funded but the user will still be charged a listing fee by the frontend.

Recommendation:

> Minimum amounts for the fundTarget and endBlock could be added to prevent immediately unusable/ unintended campaigns. Service fee is already capped at 0.01BCH so no risk of large user loss from a malicious frontend.

Status: Acknowledged

Project Response:

Minimal risk of any user loss. Will add more contract-level checks to campaign creation on a future version of the contracts.

LIFESTONE
LABS

🌐 lifestonelabs.com

✉ info@lifestonelabs.com