



SMART CONTRACT SECURITY AUDIT

# Fuelup.fun

Launchpad Contracts

CLIENT

Fuelup.fun

PLATFORM

Fuel

LANGUAGE

Sway

REVIEW

October 2024

## DOCUMENT

## Document information

|                       |                                                                                                              |
|-----------------------|--------------------------------------------------------------------------------------------------------------|
| Project               | Fuelup.fun, Launchpad Contracts                                                                              |
| Client                | Fuelup.fun                                                                                                   |
| Project URL           | <a href="https://fuelup.fun/">https://fuelup.fun/</a>                                                        |
| Platform              | Fuel                                                                                                         |
| Language              | Sway                                                                                                         |
| Repository            | <a href="https://github.com/LYNC-WORLD/fuelup-fun-smart-contracts">LYNC-WORLD/fuelup-fun-smart-contracts</a> |
| Commit                | Not recorded in the original report                                                                          |
| Methods               | Manual Analysis, Functional Testing, Automated Testing                                                       |
| First review          | 18 to 25 October 2024                                                                                        |
| Updated code received | 18 October 2024                                                                                              |
| Second review         | 25 October 2024                                                                                              |

## Document history

| VERSION | DATE         | CHANGE                                                                |
|---------|--------------|-----------------------------------------------------------------------|
| 1.0     | October 2024 | Original report issued <a href="#">View original report</a>           |
| 1.1     | October 2026 | Reformatted into the Safe Edges standard template. Findings unchanged |

This report was reformatted into the Safe Edges standard template in October 2026. Findings, severities, recommendations and code references are unchanged from the original report, which is preserved unchanged in the [Safe Edges Audit Portfolio on GitHub](#).

## CONTENTS

# Contents

---

|    |                                                      |    |
|----|------------------------------------------------------|----|
| 01 | Executive summary                                    | 3  |
| 02 | Scope                                                | 4  |
| 03 | Methodology                                          | 5  |
| 04 | Severity and status                                  | 6  |
| 05 | Findings summary                                     | 7  |
| 06 | Detailed findings                                    | 8  |
|    | M-01 Asset Reserves Update Mechanism                 | 8  |
|    | M-02 Uncapped Fee Percentage                         | 8  |
|    | M-03 Missing Reentrancy Protection                   | 9  |
|    | L-01 Fee Amount Validation                           | 9  |
|    | L-02 Missing Zero Address Check in Withdraw Function | 10 |
|    | I-01 Unused Timestamp Import                         | 10 |
| 07 | Automated tests                                      | 11 |
| A  | Appendix: vulnerability checklist                    | 12 |
|    | Disclaimer                                           | 13 |

# Executive summary

## About Fuelup.fun

Fuel Up is a meme coin launchpad and marketplace for fair buying, selling, and trade of meme coins. Assets Fuel Up Network is 100% safe and secure with no pre-mint sale or team allocation to prevent rug pulls.

## Summary of the review

Safe Edges reviewed the Fuelup.fun launchpad contracts on Fuel between 18 and 25 October 2024. Six issues were identified: three Medium, two Low and one Informational. The team fixed all six, and the fixes were confirmed in a second review on 25 October 2024.

## Findings by severity



## Findings by status

| STATUS   | ISSUES |
|----------|--------|
| Resolved | 6      |

## Scope

---

|            |                                       |
|------------|---------------------------------------|
| Repository | LYNC-WORLD/fuelup-fun-smart-contracts |
| Commit     | Not recorded in the original report   |
| Platform   | Fuel                                  |
| Language   | Sway                                  |

Only the code at the commit above was reviewed. Changes made after this commit are not covered by this report.

## Methodology

---

Methods recorded for this engagement: Manual Analysis, Functional Testing, Automated Testing.

### Manual review

Auditors read the in-scope code line by line, tracing control flow, state changes and trust boundaries against the intended behaviour described by the client.

### Threat modelling

We map privileged roles, external dependencies and value flows, then test the assumptions the code makes about each of them.

### Functional and automated testing

Where tooling exists for the language and chain, we run static analysis and targeted tests to confirm suspected issues and check edge cases.

### Fix review

After the client addresses the findings, we review the updated code and record the final status of each issue.

## Severity and status

### Severity levels

| SEVERITY      | DEFINITION                                                                                                                                            |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| Critical      | Direct loss or permanent lock of funds, or full compromise of the protocol, with few or no preconditions. Must be fixed before deployment.            |
| High          | Significant loss of funds, broken core functionality or manipulation of critical state under realistic conditions. Should be fixed before deployment. |
| Medium        | Limited loss or degraded functionality, or issues that need specific conditions or privileged access to exploit. Should be fixed.                     |
| Low           | Minor impact, low likelihood, or deviations from best practice. Fix when practical.                                                                   |
| Informational | Code quality, documentation, gas and style observations with no direct security impact.                                                               |

### Status definitions

| STATUS             | DEFINITION                                                             |
|--------------------|------------------------------------------------------------------------|
| Resolved           | The issue was fixed and the fix was verified.                          |
| Partially resolved | The risk was reduced but not fully removed.                            |
| Acknowledged       | The client accepted the issue without changing the code.               |
| Open               | The issue was not addressed at the time of the report.                 |
| Not recorded       | The original report did not record a resolution status for this issue. |

## Findings summary

| ID   | TITLE                                           | SEVERITY      | STATUS     |
|------|-------------------------------------------------|---------------|------------|
| M-01 | Asset Reserves Update Mechanism                 | Medium        | ● Resolved |
| M-02 | Uncapped Fee Percentage                         | Medium        | ● Resolved |
| M-03 | Missing Reentrancy Protection                   | Medium        | ● Resolved |
| L-01 | Fee Amount Validation                           | Low           | ● Resolved |
| L-02 | Missing Zero Address Check in Withdraw Function | Low           | ● Resolved |
| I-01 | Unused Timestamp Import                         | Informational | ● Resolved |

## Detailed findings

### M-01 Asset Reserves Update Mechanism

Severity **Medium** Status ● Resolved

#### DESCRIPTION

The **remain\_asset\_reserves** function lacks a proper update mechanism. Once the contract is live, it cannot adjust or update funds, which may lead to significant calculation errors and operational impacts.

#### IMPACT

Inaccurate reserve tracking over time could compromise the integrity of the protocol.

```
remain_asset_reserves: 0
```

#### RECOMMENDATION

- Implement a proper update function for reserves.
- Add validation checks for reserve updates.

### M-02 Uncapped Fee Percentage

Severity **Medium** Status ● Resolved

#### DESCRIPTION

The fee percentage only checks for values less than or equal to 100 but lacks a reasonable upper limit. This could potentially allow the system to charge users a 100% fee, resulting in users being unable to withdraw funds. The absence of a defined upper limit may lead to centralization and power concentration among those who can afford high fees, creating additional security risks. It is crucial for the system to properly define an upper limit on fees to protect user funds and maintain trust in the contract.

```
require(cfg.fee_percent <= 100, "fee-percent-cannot-exceed-100");
storage.cfg.write(cfg);
initialize_ownership(owner);
```

#### IMPACT

The protocol can charge 100% of user funds as fees.

## RECOMMENDATION

```
require(cfg.fee_percent <= 30, "Fee percentage cannot exceed 30");
```

Add Additional checks

### M-03 Missing Reentrancy Protection

Severity **Medium** Status ● Resolved

## DESCRIPTION

There is a lack of reentrancy guards in functions that perform external calls. The code does not consistently follow the Checks-Effects-Interactions (CEI) pattern.

## IMPACT

this vulnerability could lead to reentrancy attacks, allowing attackers to drain funds from the contract.

## RECOMMENDATION

- Implement a reentrancy guard modifier.
- Follow the CEI pattern strictly.

### L-01 Fee Amount Validation

Severity **Low** Status ● Resolved

## DESCRIPTION

There is no validation for zero fee amounts before transfer.

```
let fee_amount = (eth_amount * fee_percent) / 100;  
transfer(fee_receiver, AssetId::base(), fee_amount);  
transfer(msg_sender().unwrap(), asset_id, amount_out);
```

## IMPACT

The code may attempt to transfer zero fees to the receiver, which could be a logical error.

## RECOMMENDATION

```
if fee_amount > 0 {  
    // Proceed with fee transfer  
}
```

## L-02 Missing Zero Address Check in Withdraw Function

Severity **Low** Status ● Resolved

### DESCRIPTION

The withdraw function does not validate the recipient address, potentially allowing transfers to the zero address.

```
#[storage(read, write)]
fn withdraw_assets_from_pool(pool_asset: AssetId, to: Identity){
```

### IMPACT

This could result in permanent loss of funds if assets are transferred to the zero address.

### RECOMMENDATION

```
fn withdraw(to: Address, amount: u64) {
  require(to != Address::from(zero), "Cannot withdraw to zero address");
  // Rest of the function
}
```

## I-01 Unused Timestamp Import

Severity **Informational** Status ● Resolved

### DESCRIPTION

There is an unused import of `block::timestamp`.

While there is no security impact, this affects code cleanliness and readability.

## Automated tests

---

No major issues were found. Some false positive errors were reported by the tools. All the other issues have been categorized above according to their level of severity.

## APPENDIX A

# Vulnerability checklist

Vulnerability classes checked during this review, as listed in the original report.

|                                       |                                       |
|---------------------------------------|---------------------------------------|
| ✓ Arbitrary write to storage          | ✓ Upgradeable safety                  |
| ✓ Centralization of control           | ✓ Using inline assembly               |
| ✓ Ether theft                         | ✓ Style guide violation               |
| ✓ Improper or missing events          | ✓ Parallel Execution safety           |
| ✓ Logical issues and flaws            | ✓ UTXO Model Verification             |
| ✓ Arithmetic Computations Correctness | ✓ FuelVM Opcodes                      |
| ✓ Race conditions/front running       | ✓ Cross-Chain Interactions            |
| ✓ Re-entrancy                         | ✓ Modular Design                      |
| ✓ Malicious libraries                 | ✓ Access Control Vulnerabilities      |
| ✓ Address hardcoded                   | ✓ Denial of Service (DoS)             |
| ✓ Divide before multiply              | ✓ Oracle Manipulation                 |
| ✓ Integer overflow/underflow          | ✓ Signature Replay Attacks            |
| ✓ ERC's conformance                   | ✓ Improper Handling of External Calls |
| ✓ Missing Zero Address Validation     | ✓ Proxy Storage Collision             |
| ✓ Revert/require functions            | ✓ Use of Deprecated Functions         |

# Disclaimer

---

This report is based on the code at the commit listed in the Scope section and on information the client provided during the review. It reflects Safe Edges' assessment at the time of the review and is not a guarantee that the code is free of vulnerabilities.

Changes made after the reviewed commit, deployment configuration, off-chain components and third-party dependencies are outside the scope of this report unless stated otherwise.

Security is ongoing. We recommend continued testing, monitoring and bug bounty coverage alongside any audit.

This report is not investment advice and is not an endorsement of the project, its team or any token.



Safe Edges is a Web3 security firm and official security partner for the Fuel, ENI and XI EcoChain ecosystems. We work across smart contract audits, Rust and WASM reviews, ZK circuit review, AI and MCP security, and penetration testing.

200+

PROJECTS AUDITED

\$4B+

TVL SECURED

0

POST-AUDIT HACKS