Coinsult

Advanced Manual Smart Contract Audit

Project: Apollo Ventures (LaunchpadPresale.sol)
Website: https://a11.ventures

🟢 Low-risk	🟡 Medium-risk	🔴 High-risk
4 low-risk code issues found	0 medium-risk code issues found	0 high-risk code issues found

Contract address
(LaunchpadPresale.sol) Not deployed yet

Disclaimer

Coinsult is not responsible if a project turns out to be a scam, rug-pull or honeypot. We only provide a detailed analysis for your own research.

Coinsult is not responsible for any financial losses. Nothing in this contract audit is financial advice, please do your own research.

The information provided in this audit is for informational purposes only and should not be considered investment advice. Coinsult does not endorse, recommend, support or suggest to invest in any project.

Coinsult can not be held responsible for when a project turns out to be a rug-pull, honeypot or scam.

Tokenomics

Not deployed yet

Source code

Coinsult was commissioned by Apollo to perform an audit based on the following smart contract:

https://github.com/infocodiste/apolloVentures/blob/master/contracts/launchpad/LaunchpadPresale.sol

Note: This project uses openzeppelin imports. While we do check the full contract for vulnerabilities at the time of the audit, we can not ensure the correctness of these imported modules.

Manual Code Review

🟢 Low-risk

4 low-risk code issues found.

Could be fixed, will not bring problems.

- Consider using .div() and .mul() for dividing and multiplications

```
// Adding Liquidity
uint256 amountToLiquidity = (address(this).balance *
par.liquidityPercent) / 10000;
uint256 tokensToLiquidity = (par.listingRate * amountToLiquidity) / 1
ether;
```

- Owner can pause and unpause the contract

```
    function pause() external onlyOwner {
        _pause();
    }

    function unpause() external onlyOwner {
        _unpause();
    }
```

- Unclear or inaccurate comment: `// 259200 == 72 hours`

```
        if ((block.timestamp - par.endTime) < 300) {    // 259200 == 72
hours
            require(msg.sender == owner(), "Not Owner");
        }
```

- Block.timestamp can be manipulated by miners.

  Avoid relying on block.timestamp.

  More information:
  https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

```
    function unlockLiquidity() external onlyOwner {
        require(block.timestamp > liqidityLockedTill, "Liquidity
Locking Period is not Over");

        IUniswapV2Factory factory =
IUniswapV2Factory(router.factory());

        address pair = factory.getPair(address(token), router.WETH());
        uint256 LPBalance = IERC20(pair).balanceOf(address(this));

        IERC20(pair).safeTransfer(owner(), LPBalance);
    }
```

🟡 Medium-risk

0 medium-risk code issues found.
Should be fixed, could bring problems.

🔴 High-risk

0 high-risk code issues found
Must be fixed, and will bring problems.

Extra notes by the team

Note: This project uses openzeppelin imports. While we do check the full contract for vulnerabilities at the time of the audit, we can not ensure the correctness of these imported modules.

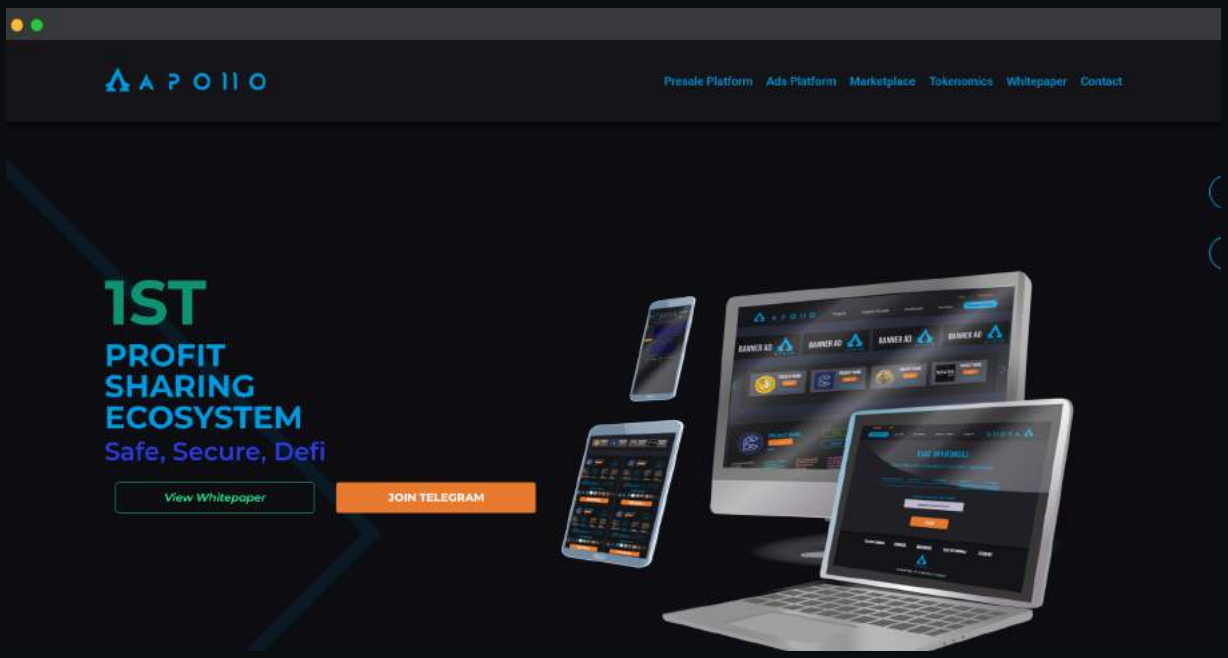
🟡 Dev notes can be deleted upon deployment

🔴 Owner can pause and unpause the contract

Contract Snapshot

```
contract LaunchpadPresale is Initializable, ReentrancyGuard, Ownable,
Pausable {
    using SafeERC20 for IERC20;

    address public feeAddress;

    IERC20 public token;
    IUniswapV2Router02 public router;
    IStaking public stakingContract;

    uint256 public totalBought;
    uint256 public totalClaimed;
    uint256 public liqidityLockedTill;
    uint256 public influencerFee;
    uint256 public raisedFee;

    bool public fail;
    bool public isFinalized;

    struct Parameters {
        uint256 presaleRate;
        uint256 softcap;
        uint256 hardcap;
        uint256 minBuy;
        uint256 maxBuy;
        uint256 startTime;
        uint256 endTime;
        uint256 liquidityPercent;
        uint256 lockingPeriod;
        uint256 listingRate;
    }

    struct VestingParameters {
        bool vesting;
        uint256 vestingFirstRelease;  // In percentage multiplied with
100
        uint256 vestingCyclePeriod;
        uint256 vestingCycleRelease;  // In percentage multiplied with
100
        uint256 vestingCycles;
```

Website Review



Coinsult checks the website completely manually and looks for visual, technical and textual errors. We also look at the security, speed and accessibility of the website. In short, a complete check to see if the website meets the current standard of the web development industry.

- Mobile Friendly
- Contains no jQuery errors
- SSL Secured
- No major spelling errors

Loading speed: 86%

Rug-pull Review

Based on the available information analyzed by us, we come to the following conclusions:

🟢 Locked Liquidity - Not applicable

🟢 Large unlocked wallets - Not applicable

🟢 Doxxed Team

Honeypot Review

Based on the available information analyzed by us, we come to the following conclusions:

🟢 Ability to sell - Not applicable

🔴 Owner is able to pause the contract

🟢 Router not hard coded in the contract - Not applicable

Note: Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by the project owner.