GoEthMe Documentation

The GoEthMe Contract is a unique decentralized Crowdfund contract designed for members of a DAO to create fundraising campaigns around Wildlife Conservation programs, This contract enables the creation of campaigns, contribution and management of funds contributed to the campaigns. Campaigns proposed are subject to a vote by members of the DAO. The contract issues a Reward NFT to individuals who contribute to approved campaigns.

Overview

The GoEthMe contract comprises the following key components

ERC721 Implementation: This contract extends the functionality of the ERC721 contract and IERC721 interface, providing the structure for minting and issuing the Rewards NFT.

GoFund Struct: This Struct contains the information pertaining to the Campaign created.

Contract Details

State Variables

id: A public variable that holds the ID of the campaign.

funder: A mapping that stores the campaign ID corresponding to the Campaign information.

contribute: A mapping that stores the campaign IDs to the contributor address and their contribution amounts.

hasContributed: A mapping that stores the campaign IDs to the contributor address and information on the status of their contribution.

Functions

createGoFundMe: Enables a user to create a new GoFund campaign after approval from the DAO and returns the ID of the Campaign.

contributeEth: Enables a user to contribute Ether to a GoFund campaign, then safely mints a Reward NFT to the user.

getContributedFunds: Enables the owner of a GoFund campaign to retrieve the contributed funds.

getContributors: Retrieves a list of Contributors and their contribution balances for a specified GoFund campaign.

Error Handling:

InsufficientInput: Throws an error if the contribution amount is zero or negative.

NotActive: Throws an error if a user attempts to contribute an inactive campaign.

NotActiveCause: Throws an error if the owner of a campaign attempts to withdraw from an inactive campaign.

ExceededFundingGoal: Throws an error if a contribution would exceed the campaign's funding goal.

NotInDuration: Throws an error if a user attempts to contribute to a campaign that is not within its duration.

NotOwner: Throws an error if the caller is not the owner of the campaign.

TimeNotReached: Throws an error if the owner of a campaign attempts to withdraw from a campaign that has not ended.

Usage

This contract provides a secure and efficient way to create fundraising campaigns for Conservation programs. It ensures only campaigns approved by the DAO are allowed to launch. The contract is set up to ensure ownership restrictions and prevent unauthorized withdrawals. Additionally the contract enables the storage and retrieval of contributors and campaigns information.