SafeClub
Smart Contract

Projet Blockchain & Gouvernance Décentralisée

RÉALISÉ PAR

Oumayma Naffeti & Sywar Moussa

CLASSE & MODULE

SSIR-G | Module Blockchain

ANNÉE UNIVERSITAIRE

2024–2025

TYPE DE PROJET

Rapport Final de Validation

PROJET BLOCKCHAIN

Introduction

SafeClub est une solution innovante de gestion collective de fonds sur la blockchain Ethereum, éliminant le besoin d'intermédiaires de confiance traditionnels.

Contexte Décentralisé

Gestion d'un fonds commun (trésorerie) entièrement régie par du code (Smart Contract), garantissant transparence et immuabilité.

Gouvernance Participative

Chaque dépense nécessite une proposition formelle soumise au vote des membres, validée uniquement si le quorum est atteint.

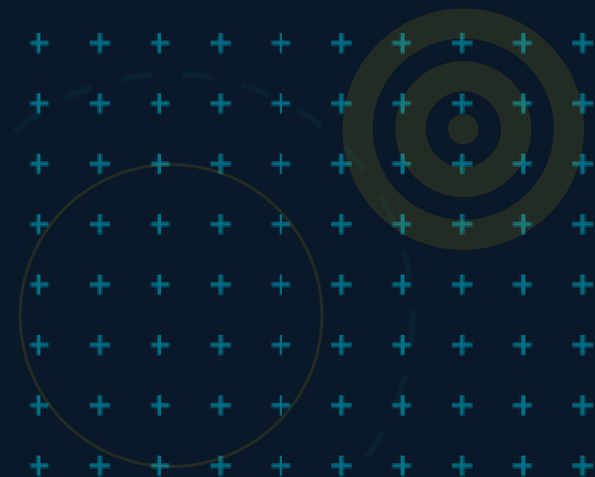
Objectifs Clés

Assurer la sécurité maximale des fonds, garantir la cohérence

VISUALISATION DU RESEAU

Objectifs du Projet

Développer une solution blockchain robuste répondant aux exigences de sécurité et de décentralisation.



Conception Sécurisée

Développement d'un smart contract en Solidity suivant les meilleures pratiques de sécurité.

Système de Gouvernance

Implémentation d'un mécanisme de propositions et de votes transparent pour les membres.

Protection des Fonds

Garantie contre les abus et les attaques (Reentrancy, double dépense) pour sécuriser la trésorerie.

Tests & Validation

Validation rigoureuse du contrat via des scénarios réalistes sur réseau de test.

Technologies & Outils

La stack technique complète utilisée pour le développement, le déploiement et la sécurisation du smart contract SafeClub.

Solidity

Langage de programmation orienté objet pour l'écriture du smart contract sur la blockchain Ethereum.

Remix IDE

Environnement de développement intégré pour la compilation, le déploiement et les tests unitaires.

MetaMask

Portefeuille numérique permettant la signature des transactions et la gestion des comptes.

Ethereum Sepolia Testnet

Réseau de test public utilisé pour valider le fonctionnement du contrat sans frais réels.

OpenZeppelin

Bibliothèque de contrats standards sécurisés, utilisée notamment pour la protection Reentrancy.

Architecture du Smart Contract

Structure modulaire et flux d'interaction

Gestion des Rôles

Owner : Déploiement, Ajout/Suppression membres

Membres : Accès aux fonctions de vote et proposition

Mapping `isMember`

CRÉATION

Propositions

Struct `Proposal`

ID, Destinataire, Montant, Description

Deadline & État (Executed)

DÉCISION

Système de Vote

1 Membre = 1 Vote

Comptage Pour / Contre

Vérification Quorum & Exécution

Gestion des Membres

Un système de contrôle d'accès strict distinguant les rôles pour assurer une gouvernance saine et sécurisée.

Rôle Owner

Responsable du déploiement initial, de l'onboarding des nouveaux participants et de la révocation des accès.

Vérification d'Accès

Utilisation du mapping `isMember` pour valider instantanément l'appartenance d'une adresse au club.

Fonctions Admin

Les fonctions `addMember` et `removeMember` sont protégées par le modificateur `onlyOwner`.

Sécurité & Privilèges

L'accès aux fonctions critiques (propositions, votes) est strictement réservé aux membres actifs.

Fonctionnalités Principales

Un ensemble complet de fonctions pour gérer la gouvernance et la trésorerie de manière autonome et sécurisée.

Gestion des Membres

Le propriétaire (Owner) contrôle l'accès en ajoutant ou supprimant des membres autorisés.

Réception d'ETH

Le smart contract agit comme un coffre-fort capable de recevoir et sécuriser les dépôts.

Création de Propositions

Les membres soumettent des dépenses avec montant, destinataire et description.

Système de Vote

Vote démocratique unique (Pour/Contre) par membre pour chaque proposition.

Exécution Sécurisée

Transfert automatique des fonds uniquement après validation du quorum et délai.

Documentation Technique

Smart Contract SafeClub.sol

Solidity 0.8.x

DONNÉES & ÉTAT

SafeClub_Storage.sol

```
// Gestion des membres
mapping(address => bool) public isMember;

// Structure d'une proposition
struct Proposal {
uint256 id;
address to;
uint256 amount;
string description;
uint256 deadline;
bool executed;
uint256 forVotes;
uint256 againstVotes;
}

// Stockage des propositions
mapping(uint256 => Proposal) public proposals;

// Suivi des votes (anti-double vote)
mapping(uint256 => mapping(address => bool)) public hasVoted;
```

FONCTIONS CLÉS

SafeClub_Logic.sol

```
// --- Gestion des Accès ---
function addMember(address _member) external onlyOwner;
function removeMember(address _member) external onlyOwner;

// --- Finance ---
function deposit() external payable;
receive() external payable;

// --- Gouvernance ---
function createProposal(
address _to,
uint256 _amount,
string calldata _desc,
uint256 _duration
) external;

function vote(uint256 _id, bool _support) external;

function execute(uint256 _id) external nonReentrant;
```

Keywords Types Functions

Sécurité du Contrat

Trois mécanismes fondamentaux pour garantir l'intégrité du système et la protection des fonds.

Protection Anti-Reentrancy

Utilisation du module **ReentrancyGuard** et respect du modèle *Checks-Effects-Interactions* pour sécuriser les transferts.

Contrôle d'Accès Strict

Restriction des fonctions critiques via les modificateurs `onlyOwner` et vérification d'appartenance `isMember`.

Validation d'État & Logique

Vérifications systématiques : quorum atteint, deadline respectée, proposition non exécutée et solde suffisant.

Tests et Validation

Méthodologie et scénarios validés

Déploiement Initial

Déploiement sur Sepolia Testnet via Remix IDE et vérification de l'adresse Owner.

Gestion des Membres

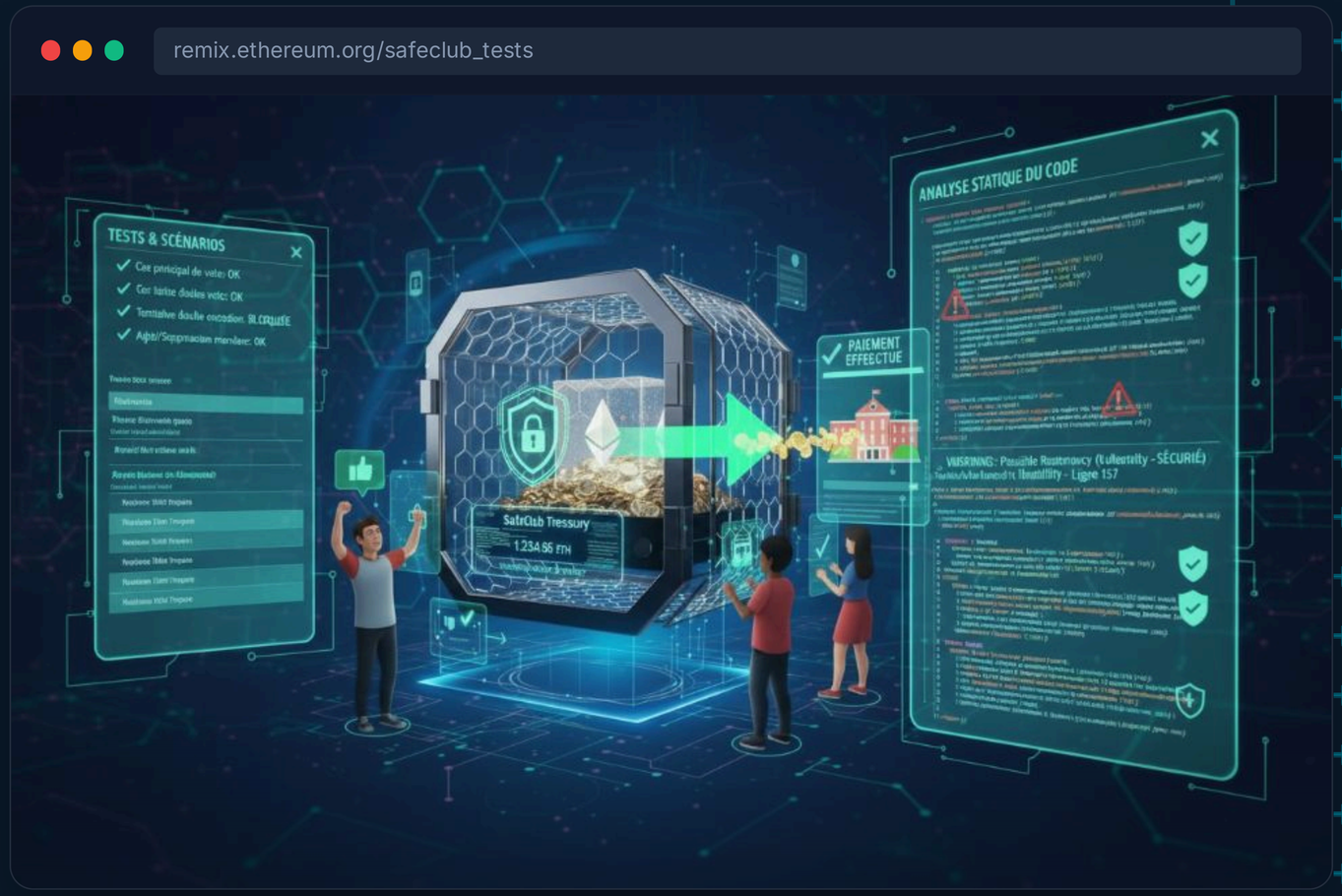
Test des fonctions `addMember` et `removeMember` avec vérification des permissions.

Cycle de Proposition

Création de propositions, dépôt d'ETH et simulation de votes multiples (pour/contre).

Exécution Sécurisée

Validation du quorum, respect de la deadline et transfert effectif des fonds.



TESTS VALIDÉS
100% Succès

Règles de Décision

Critères obligatoires pour l'exécution d'une proposition sur la Blockchain

QUORUM ATTEINT

Le nombre de votes favorables doit être supérieur au seuil défini.

```
forVotes ≥ minQuorum
```

+

PÉRIODE TERMINÉE

La date actuelle doit avoir dépassé la date limite de vote.

```
block.timestamp > deadline
```

+

ÉTAT UNIQUE

La proposition ne doit jamais avoir été exécutée auparavant.

```
!executed
```

TRANSACTION VALIDÉE

Le transfert de fonds est autorisé et exécuté.

Limites & Améliorations

Bien que fonctionnel, le projet actuel présente des contraintes qui définissent les futurs axes de développement.

⚠ LIMITES ACTUELLES

Système de vote sans délégation, interface utilisateur minimale et absence de pondération des voix.

Interface Web Dédiée (DApp)

Développement d'une interface frontend (React/Vue) connectée via Web3.js pour une meilleure expérience utilisateur.

Gouvernance Avancée

Intégration de la délégation de vote et d'un système de pondération selon l'implication des membres.

Mécanisme Multisignature

Renforcement de la sécurité pour l'exécution des transactions critiques nécessitant plusieurs signatures.

Tokens de Gouvernance

Création d'un token ERC-20 pour gérer les droits de vote et automatiser la distribution des parts.

Conclusion & Perspectives

Synthèse du projet SafeClub et prochaines étapes

Réalisations Clés

Gouvernance Décentralisée

Implémentation réussie d'un système complet de propositions et de votes on-chain.

Sécurité Robuste

Protection contre les attaques (Reentrancy) et gestion stricte des permissions Owner/Member.

Validation Technique

Tests validés sur Sepolia Testnet confirmant la fiabilité du smart contract.

Pistes d'Évolution

Interface DApp Web3

Développement d'une interface utilisateur (React/Next.js) pour faciliter l'interaction.

Tokenisation (ERC-20)

Introduction de tokens de gouvernance pour un système de vote pondéré.

Sécurité Avancée

Intégration de mécanismes de multisignature pour l'approbation des grosses dépenses.

PROJET CONFORME & VALIDÉ