BLOCKCHAIN

Activities – 5

Rapport de Projet : Mini Réseau Social en Solidity

Realized by:

ABDELALI IBN TABET

Introduction

Dans le cadre de cet atelier, nous avons développé un smart contract en Solidity pour simuler un mini réseau social. Ce réseau permet aux utilisateurs de publier des messages, de consulter les messages existants, et de vérifier le nombre total de publications. Ce projet a été implémenté sur Ethereum via le réseau de test SepoliaETH.

Objectifs

Les objectifs de ce projet sont :

- Créer un smart contract avec des fonctionnalités basiques de réseau social.

- Déployer et tester le smart contract sur un réseau de test Ethereum : SepoliaETH.

- Interagir avec le smart contract pour publier et récupérer des messages, tout en validant le fonctionnement de chaque fonctionnalité.

Outils Utilisés

- Remix IDE : Pour écrire et déployer le smart contract.

- MetaMask : Pour interagir avec le réseau de test Sepolia et exécuter des transactions.

- Réseau de test SepoliaETH.

Conception du Smart Contract

Structure du Code

Le smart contract est structuré de manière à permettre :

- La publication de messages avec l'adresse de l'auteur.

- La consultation de messages spécifiques.

- La récupération du nombre total de messages publiés.

Explication des Fonctions

publishPost : Cette fonction permet aux utilisateurs de publier un message. Elle enregistre le message et l'adresse de l'utilisateur qui appelle la fonction dans le tableau posts.

getPost : Permet de récupérer un message à un index donné. Retourne le contenu du message et l'adresse de l'auteur.

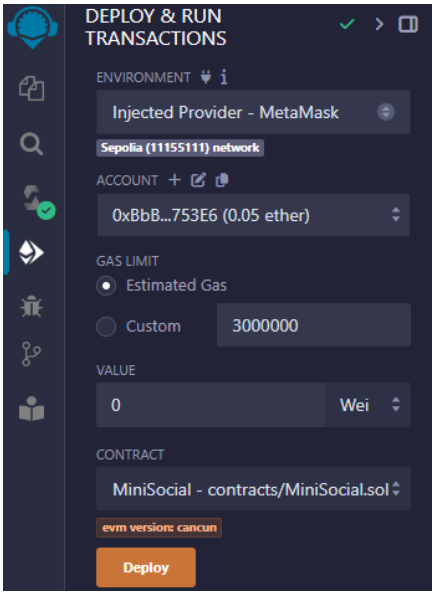
getTotalPosts : Retourne le nombre total de messages publiés dans le réseau.

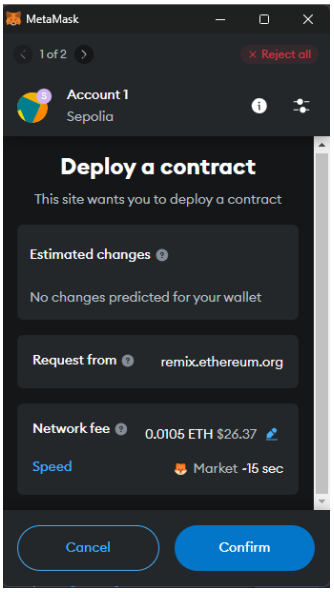
Déploiement

On vérifie que l'extension MetaMask est bien installée dans le navigateur, connectée, et configurée sur le réseau de test Sepolia.

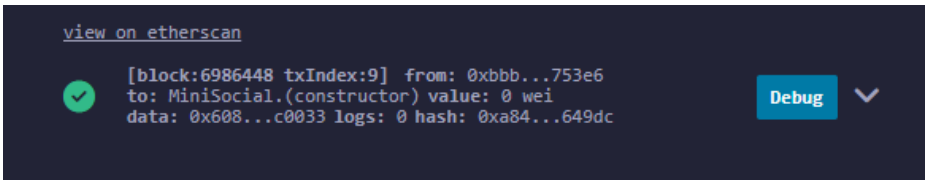
On passe à l'onglet Deploy & Run Transactions et on choisit l'environnement 'Injected Provider – MetaMask' pour se connecter au réseau Sepolia via MetaMask.

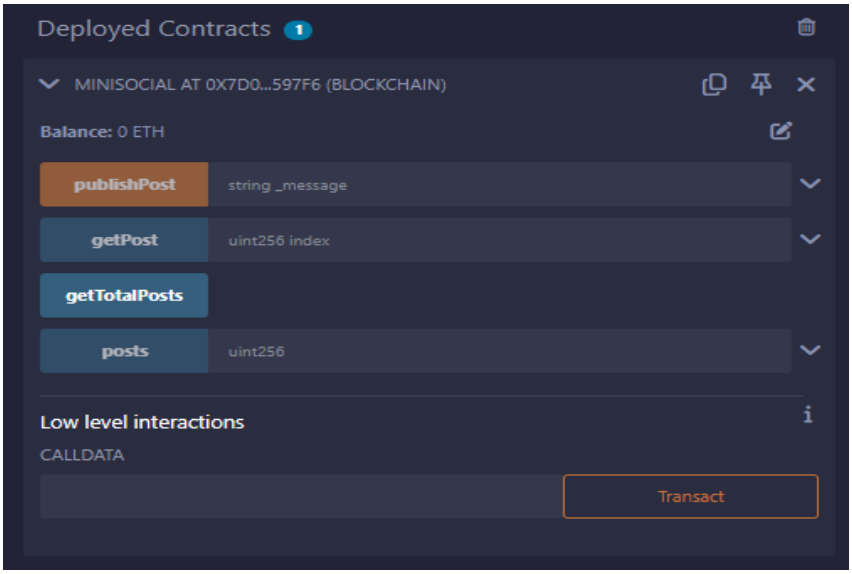
Une fenêtre MetaMask s'ouvre pour demander confirmation ; on valide la transaction de déploiement.





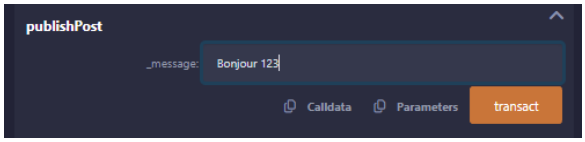
Une fois déployé, le contrat est visible dans Remix sous Deployed Contracts, avec l'adresse de contrat générée.

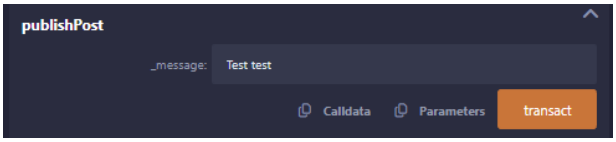




Test de la Fonction de Publication

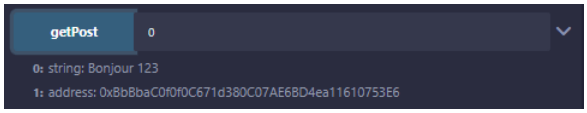
On utilise la fonction publishPost dans Remix pour publier un message en saisissant une chaîne de caractères dans le champ _message et en cliquant sur transact.

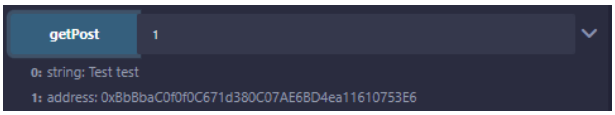




Test de la Fonction de Consultation

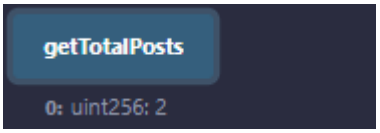
Pour vérifier les messages publiés, on utilise la fonction getPost, en entrant l'index du message souhaité (ex. 0 pour le premier message). Remix retourne le message et l'adresse de l'auteur associé à cet index, permettant de confirmer que les messages sont bien stockés.





Vérification du Nombre Total de Messages

On utilise la fonction getTotalPosts pour vérifier le nombre total de messages publiés en cliquant simplement sur call. Remix affiche le nombre total, qui devrait correspondre au nombre de messages publiés.



Conclusion

Ce projet de smart contract a permis de créer une application décentralisée de réseau social basique. Les fonctionnalités de publication, de consultation, et de comptage des messages ont été implémentées avec succès et validées par des tests. Ce projet a également permis de se familiariser avec les concepts de base de Solidity et la configuration d'un environnement de test Ethereum.