Token Faucet DApp – Complete Project Documentation

1. Project Overview

The Token Faucet DApp is a decentralized application designed to distribute ERC-20 tokens to users in a controlled and secure manner. Users can request tokens periodically, subject to cooldown periods and lifetime limits. The project demonstrates smart contract development, testing, deployment, frontend integration, and containerization using Docker.

2. Architecture Overview

The architecture consists of three main layers: Smart Contracts, Frontend Application, and Infrastructure. The smart contracts are deployed on the Sepolia Ethereum test network. The frontend interacts with these contracts using ethers.js and MetaMask. Docker is used to containerize and serve the frontend.

3. Smart Contract Design

The system contains two Solidity contracts: YourToken (ERC-20) and TokenFaucet.

1. YourToken (ERC-20): Fixed maximum supply, minting restricted to faucet contract.
2. TokenFaucet: Controls token distribution with cooldowns, lifetime limits, and pause functionality.

4. Deployed Contracts (Sepolia)

YourToken Address: 0xA7568A942Cef9841ABeAFAcDB955874487d143c3

TokenFaucet Address: 0x010d0a0fF7E985781A27D9ed7E971F54B76E08A0

5. Quick Start Guide

1. Clone the repository from GitHub.
2. Copy .env.example to .env and fill required values.
3. Run docker compose up to start the frontend.
4. Access the application at http://localhost:3000

6. Environment Configuration

SEPOLIA_RPC_URL: RPC endpoint for Sepolia network. PRIVATE_KEY: Wallet private key used for deployment. ETHERSCAN_API_KEY: Used for contract verification.

7. Testing Approach

Smart contracts were tested using Hardhat with Mocha and Chai. Tests cover successful claims, cooldown enforcement, lifetime limits, pause functionality, access control, and multi-user scenarios.

8. Security Considerations

Minting is restricted to the faucet contract. Admin-only functions are protected using require checks. Cooldown and lifetime limits prevent abuse. Sensitive data such as private keys are stored in environment variables.

9. Docker & Deployment

The frontend is containerized using Docker with a health check endpoint at /health. Docker Compose manages service configuration and port exposure.

10. Known Limitations & Future Improvements

The faucet currently supports only one network. Future improvements include multi-network support, UI enhancements, rate-limit customization, and backend analytics.

11. Conclusion

This project demonstrates a complete end-to-end blockchain DApp workflow, from contract development to deployment, frontend integration, and containerized delivery.