**Plan de Déploiement – Projet EquiToken**

**Objectif**

Structurer le développement et le déploiement technique du projet EquiToken en trois modules principaux :

1. Le token EQT (ERC-20)
2. La DAO (gouvernance décentralisée)
3. L’application web EquiToken (interface utilisateur)

**Étapes et Priorités de Déploiement**

**Étape 1 : Déploiement du token EQT**

**Objectifs :**

* Déployer le contrat EQT sur une blockchain publique (testnet, puis mainnet)
* Assurer son intégration avec les portefeuilles comme MetaMask

**Procédure :**

1. Développer et tester localement le contrat (Hardhat ou Foundry)
2. Déployer sur un **testnet** (Goerli ou Sepolia) via Hardhat ou Remix
3. Vérifier le contrat sur Etherscan
4. Une fois validé, déployer sur le **mainnet Ethereum** ou un **Layer 2 (ex. Arbitrum)**
5. Obtenir les adresses des wallets pour l’allocation initiale
6. Prévoir du gaz (ETH) pour le déploiement (payé via wallet MetaMask)

⚠️ Le contrat est **déployé sur une blockchain publique** (non sur un serveur privé)  
✅ Il devient **immuable** et accessible globalement via son adresse blockchain

**Étape 2 : Déploiement de la DAO EquiToken**

**Objectifs :**

* Permettre la gouvernance par les détenteurs de tokens
* Gérer des décisions communautaires (vote, exécution différée via Timelock)

**Procédure :**

1. Déployer le contrat de **TimelockController**
2. Déployer le contrat **Governor (DAO)** lié au token EQT et au timelock
3. Configurer les paramètres (quorum, durée de vote, délai d’exécution)
4. Tester la création et le vote de propositions sur le testnet
5. Déployer l’ensemble sur le **mainnet (même blockchain que le token EQT)**

⚠️ Le contrat de gouvernance est aussi déployé **sur la blockchain**, pas sur un serveur personnel

**Étape 3 : Déploiement de l’application web EquiToken**

**Objectifs :**

* Fournir une interface utilisateur Web3 (dashboard, portefeuille, gouvernance)

**Procédure :**

1. Développer l’application React avec intégration Web3.js ou Ethers.js
2. Connecter l’app aux contrats EQT et DAO via leurs adresses sur la blockchain
3. Afficher les données on-chain (balances, propositions DAO, résultats de votes)
4. Déployer l’app sur une **plateforme d’hébergement web** (Vercel, Netlify, OVH)

✅ L'application web est déployée sur un **serveur traditionnel** mais **interagit avec la blockchain** en lecture/écriture

**Outil recommandé : GitHub**

* Utiliser GitHub pour versionner tout le projet (smart contracts, frontend, scripts)
* Collaboration possible via pull requests, issues et branches

**Structure du dépôt recommandée :**

/equitoken

├── /contracts # Smart contracts (EQT, DAO, Timelock)

├── /scripts # Déploiement et initialisation

├── /frontend # Application web React

├── /test # Tests unitaires et d’intégration

├── /docs # Whitepaper, infographies, roadmap

└── README.md # Présentation du projet

**Schéma technique de l’architecture**

[ Utilisateur ]

↓

[ App EquiToken (hébergée sur Vercel/OVH) ]

↓ ↘

[ Smart Contract EQT ] [ DAO + Timelock ]

↓ ↑

[ Redistribution automatique des gains (ETH/USDC) ]

**Conclusion**

✅ Le **token EQT** et la **DAO** doivent être déployés **exclusivement sur une blockchain publique** (Ethereum, Arbitrum…). Ils sont **accessibles de partout et immuables**.  
✅ L’**application web**, elle, peut être hébergée sur un **serveur classique**, et ne sert que d’interface utilisateur.  
✅ Le tout est versionné et géré via **GitHub** pour un développement structuré et collaboratif.

➡️ Prochaine étape : créer le dépôt GitHub, tester les déploiements sur testnet, puis publier progressivement en production.