Documentation technique

Cette documentation présente le fonctionnement de l'application décentralisée (DApp) `MonkeyVotingDapp`.

Stack technique

La stack technique diverge suivant l'environnement d'exécution. Ainsi, pour chaque composant nous distinguerons l'environnement par ces badges:

- env dev environnement de développement
- env test déploiement sur un réseau de test
- env prod déploiement sur un réseau de production

Truffle

env dev env test env prod

Ce composant permet le déploiement des contrats solidity sur la blockchain. Pour ce faire, son utilisation conduit à la génération d'une Application Binary Interface (ABI). Cette dernière apporte une façon:

- d'encoder les appels de contrats solidity pour l'Ethereum Virtual Machien (EVM)
- de lire la données issue des transactions

Pour ce faire, cette interface permet l'interaction entre les composants au travers d'un fichier JSON qui contient:

- une représentation des contrats
- les informations sur le réseau utilisé

Ce fichier JSON est généré dans `src/app/contracts`.

Cet élément est primordial car il nous permet, avec l'aide de web3.js, d'aider à la résolution de l'addresse du contrat dans la blockchain actuelle. En effet, puisque plusieurs réseau de déploiement sont possibles, il faut pouvoir avoir un moyen de récupérer facilement cette information.

Dans ce projet, la fonction `initElectionContract()` du *contract service* est en charge de ce déploiement.

Ganache

env dev

Ganache offre une blockchain Ethereum personnelle. Il permet ainsi d'avoir un environnement de développement plus souple. Il est possible d'importer le workspace Truffle pour tester notre application sans

avoir à la publier sur un vrai réseau décentralisé.

Ce composant apporte également la possibilité de visualiser l'état sur smart contract et l'état des différentes variables au travers d'une interface graphique.

Geth

env test env prod

Go Ethereum (Geth) est une implémentation du protocol Ethereum. Il est ainsi possible d'être un noeud dans le réseau Ethereum.

Ce composant est utilisé afin de publier les *smart contracts* sur le réseau Goerli test net.

Utilisé dans le cadre d'un déploiement vers un réseau de test, Geth offre également la possibilité de publier le blockchain sur un réseau principal.

Swarm

env test env prod

Swarm est une plateforme de stockage distribuée. Ce composant est utilisé afin de stocker la webapp sur la blockchain Ethereum.

web3.js

env dev env test env prod

Cette collection de librairies Javascript permet l'interaction avec le contrat.

Elle requiert l'ABI du contrat généré avec Truffle. De plus, l'addresse du contrat récupéré avec Truffle doit lui être fournie.

Ce composant permet d'exposer toutes les méthodes du contrat solidity.

web3.js assure également la gestion des autorisation pour les transactions avec MetaMask. Entre autre, l'envoi de requête et le calcul du coût du gas est géré par cette librarie.

MetaMask

env dev

MetaMask est un wallet Ethereum. Dans ce projet, il est utilisé en développement afin d'accéder à la Daap au travers de plusieurs identitées. Ce composant permet de tester le vote avec des comptes différents.

Solidity

env dev env test env prod

Le langage Solidity est utilisé afin de créér le *smart contract*.

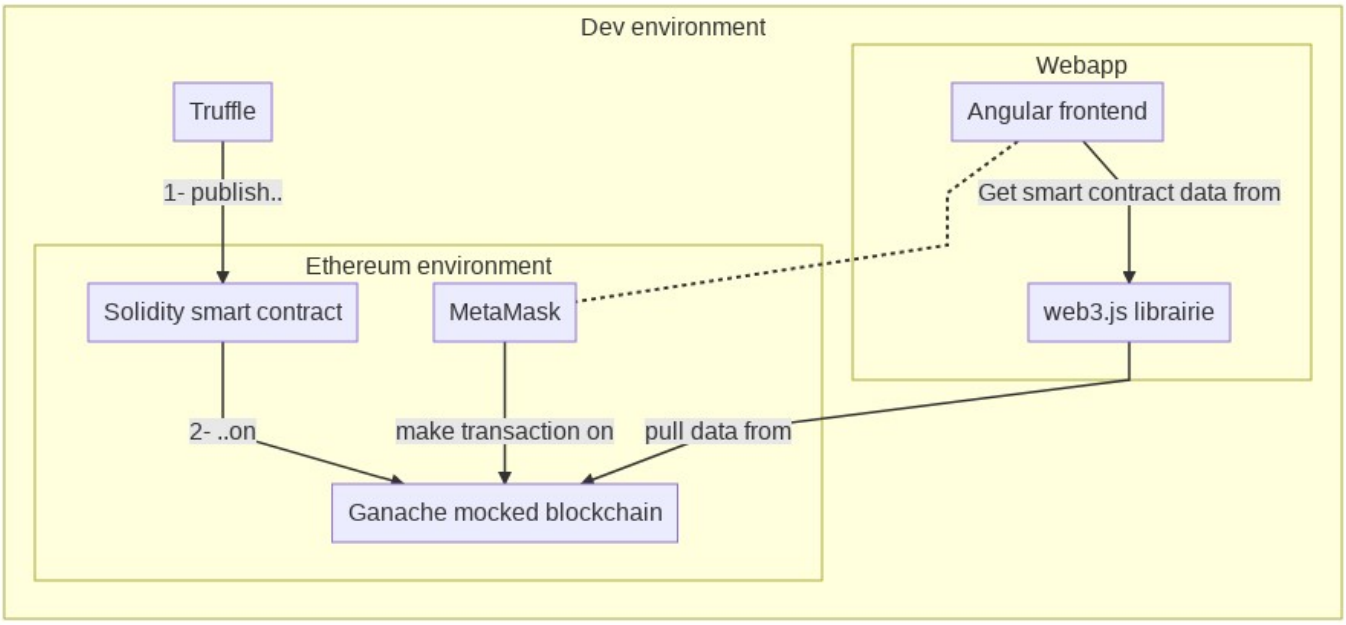
Angular

env dev env test env prod

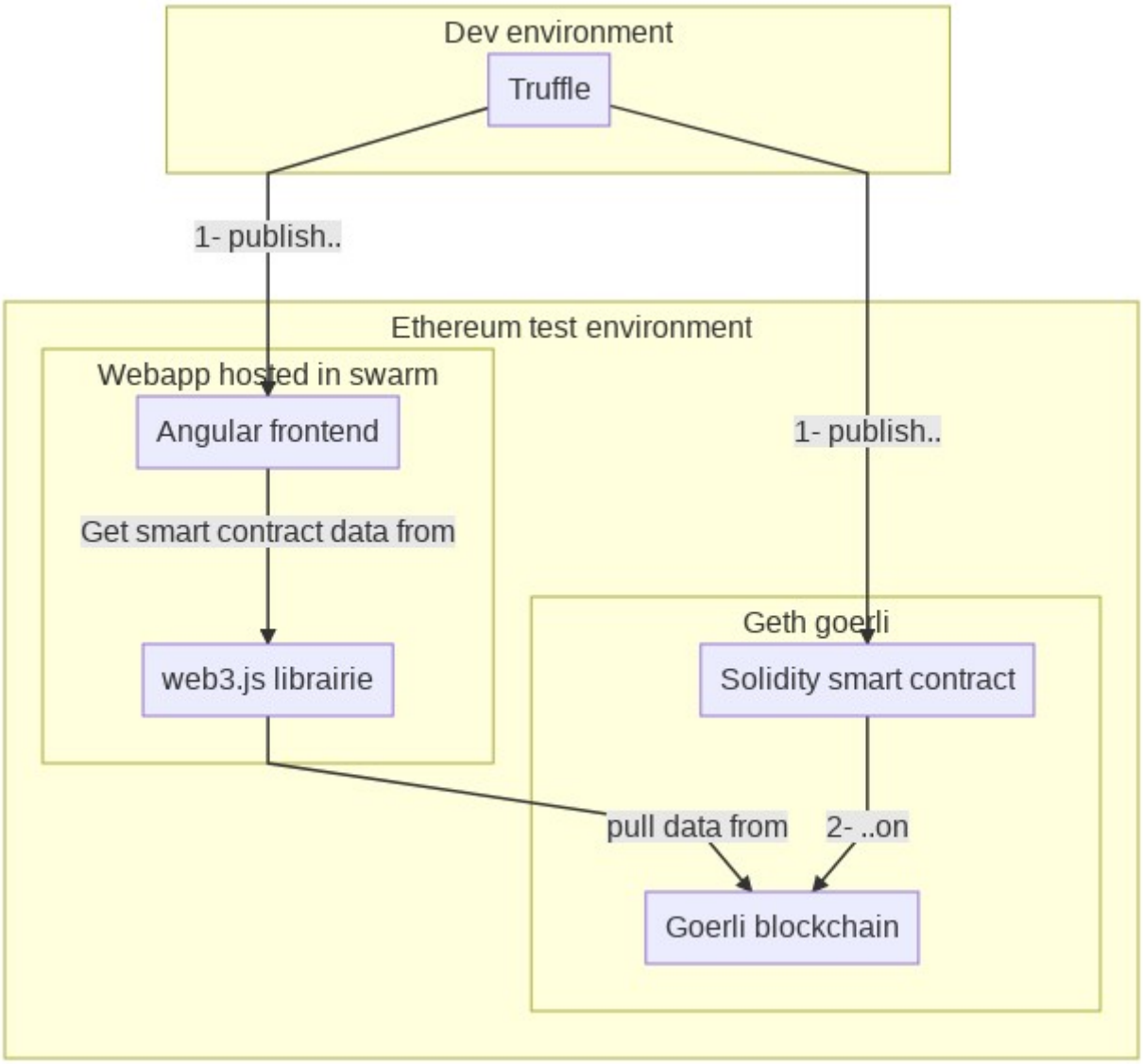
Angular est utilisé pour créér la webapp. Son adhérence au TypeScript permet de bénéficier de toutes les qualités liées à ce langage. C'est une excellente solution pour réaliser du frontend.

Architecture technique

Environnement de developpement



Environnement de test



Architecture logicielle

La webapp a été découpée en services. On en trouve 2:

- `web3.service.ts`
  - Gère la recherche du fournisseur *web3*
- `contract.service.ts`
  - Gère l'ensemble des interactions avec le *smart contract*
    - initilisation du contrat
    - gestion de toutes les interactions avec l'application