

DISTRIBUTED FACTORY

Exercice 1

Créer un wallet Ethereum web



FONCTIONNALITÉS

En gros les fonctionnalités de base de MetaMask

- ◇ Afficher les 12 mots (mnemonics) pour générer un HDWallet.
- ◇ Faire valider les 12 mots à l'utilisateur, et lui demander un mot de passe.
- ◇ Générer le HDWallet, puis en dériver une clef privée à partir du chemin (m/44'/60'/1'/0/0).
- ◇ Encrypter la clef privée avec un mot de passe (entré par l'utilisateur précédemment).
- ◇ Stocker la clef privée encryptée (nommée keystore) dans le localStorage
- ◇ Faire signer une transaction avec le wallet (décrypte la clef avec le pwd et signe la tx).
- ◇ Afficher l'adresse et la balance courante de l'utilisateur.

ANGULAR

FRONT-END

DESIGN SYSTEM

STATE
MANAGEMENT

PROVIDER
ETHEREUM

NOEUD
ETHEREUM

- ◇ Créer un projet avec [@angular/cli](https://angular.io/cli)
- ◇ Ajouter un module « wallet » avec le routing
- ◇ Créer deux dossiers dans wallet : « components » et « containers »
 - Containers : les différentes pages que verra l'utilisateur
 - Components: les sous-components (Input / Output)
- ◇ Dans containers ajouter ces 4 composants :
 - mnemonic: page pour générer les 12 mots.
 - verification: page de vérification des 12 mots + mot de passe.
 - wallet: page avec l'adresse d'affichée et sa balance en ether.
 - transaction : page qui apparait quand on veut signer une tx.
- ◇ Ajouter les composants dans le routing

ANGULAR MATERIAL

FRONT-END

DESIGN SYSTEM

STATE
MANAGEMENT

PROVIDER
ETHEREUM

NOEUD
ETHEREUM

- ◇ Ajouter au project avec la commande « [ng add @angular/material](#) ».
- ◇ Ajouter la librairie [@angular/flex-layout](#) (pour le flexbox).
- ◇ Créer un module « UI » dans lequel vous [importez et exportez les modules](#) graphiques dont vous avez besoin.
Puis importez UiModule dans WalletModule.
- ◇ Avoir une version responsive pour la validation des mots.

The desktop version of the validation page features a dark blue header. Below it is a light gray input field. Further down, three labels 'N° 2', 'N° 9', and 'N° 12' are positioned above their respective light gray input fields. At the bottom, there is a 'Mot de passe' label above a light gray input field, and a dark gray 'validate' button to its right.

Page « validation » desktop



The mobile versions of the validation page are shown in two columns. The left column represents 'Step 2/4' and features a dark blue header, a light gray input field, the label 'N° 9' above another light gray input field, and at the bottom, a dark gray left arrow, the text 'Step 2/4', and a dark gray right arrow. The right column represents 'Step 4/4' and features a dark blue header, a light gray input field, the label 'Mot de passe' above another light gray input field, a dark gray 'validate' button below it, and at the bottom, a dark gray left arrow, the text 'Step 4/4', and a dark gray right arrow.

Page « validation » Mobile

AKITA

FRONT-END

DESIGN SYSTEM

STATE
MANAGEMENT

PROVIDER
ETHEREUM

NOEUD
ETHEREUM

- ◇ Avec la [CLI de AKITA](#) :
 - Créer un [store](#).
 - Ajouter le dans le module UI « +state » dans le module.
- ◇ Utiliser les [devstools](#) avec [l'addon](#) Firefox / [extension](#) chrome pour redux
- ◇ Le state du store doit avoir :
 - L'adresse du wallet (chargée depuis le localStorage).
 - Le keystore (clef privée encryptée chargée depuis le localStorage).
 - La balance en ether (voir ethers.js).
- ◇ Le service permet de:
 - Générer des mnemonics.
 - Créer un Wallet à partir des mnemonics.
 - Encrypter / décrypter de la clef privée avec un mot de passe.
 - Getter et setter dans le localStorage.
 - Récupérer la balance courante de l'adresse.
 - > Ces fonctionnalités utilisent [ethers.js](#)

ETHERS

FRONT-END

DESIGN SYSTEM

STATE
MANAGEMENT

PROVIDER
ETHEREUM

NOEUD
ETHEREUM

- ◇ Suivre ce [tuto](#) pour permettre à angular d'utiliser « crypto »
- ◇ Utiliser cette [solution](#) pour générer les mnemonics.
- ◇ Fonction pour créer un [Wallet](#) à partir des mnemonics
- ◇ Fonction pour [encrypter](#) la clef privée avec le mot de passe
- ◇ Fonction stocker le keystore dans le [localstorage](#).
- ◇ Fonction pour [decrypter](#) le keystore avec le mot passe
- ◇ Fonction pour [signer](#) une transaction.
- ◇ Fonction pour récupérer la [balance courante](#).

Note : Toutes ces fonctions sont à mettre dans le service.

INFURA

FRONT-END

DESIGN SYSTEM

STATE
MANAGEMENT

PROVIDER
ETHEREUM

NOEUD
ETHEREUM

Pas besoin de monter un nœud Ethereum sur sa machine.
Utiliser la connexion Infura [proposé de base par ethers.js](#).
Se connecter à Kovan.