Next Phase:

目的: Monasアルファ版をリリースするためにどう実装していくのかを理想と現実のギャップを埋めながら行うことを決めたい。つまり、アルファ版でどこまで実装を行うのか決めたい

決めたい事

1. Monasサーバーで実行を行う or Peer側で実行を行う
    a. 暗号化処理をサーバーで行うか、Peer側で行えるようにするか
        i. スピード感を考えると前者の方が早く実際に使用可能にする事ができる
        ii. 後者は1つMonasで実現させなければならないことではある
            1. いずれ行うことではあり、直近で行うのか、1つ遅らせるのかの違いかと思う
2. Rustへの移行をいつ行うか
    a. 上記に繋がる話ではあるが、Rustへの移行は行いたいと思う
    b. ブラウザ処理を行う際にはWebAssemblyで行うことができる
        i. サーバー処理を行うタイミングというよりは、ブラウザ処理や実際にユーザーがソフトウェアをインストールするタイミングで実装に取り掛かる形が良いと思う
        ii. つまり、ブラウザ実行可能な状態でリリースを行うのか、一旦はサーバー処理にするのか
3. 通信はpush protocolのままか、このタイミングで移行するか
    a. セキュリティーの観点からいずれはPush protocolを外したいと考えている
    b. アルファ版を一番初めに出す際に外すかどうか？を考えるとどの通信プロトコルを使用するのかなどを技術の比較検討をおこなってから実装したいと考えているため一旦は置いておくのがいいかな？って思っている
4. DIDの認証について
    a. DIDによる制御はアルファー版で実装したいと考えている
        i. そのためにDID methodは何にするのかの検討は必要である
            1. DID method自体を作成することを考えてはいるが、それをアルファ版で公開するというよりは、時間をかける必要があると思う
            2. つまり、今回はDID methodで何が最善なのかを決めることが必要
                a. Monasに必要な要素を書き足していくと共にDIDの問題を見つけ、最強なDID methodを探る(Yudai)
5. どの機能まで実装を行うのか
    a. 現状Readのアクセス制御は可能になっている
        i. 鍵の保存場所はPush protocolを使用しているため、ある意味ではIPFS上に保存されているが、相手側のKey registry(仮)の空間内に保存可能にすることができれば、Push protocolへの依存性はなくなる可能性がありそう
            1. 共有された鍵の保存のためのレジストリをデフォルトで作成可能にする的な？
        ii. で、それには書き込みアクセス制御が必要になるのではないか？
            1. 書き込みアクセス制御は複数人で空間の制御を可能にするかどうかの話と同等だと思う

            a. コレには誰が可能なのかのリストの保持が必要になる
            b. ソーシャルグラフ的なのをどこに保存するのかに近い
                i. 楽観的すぎるけどMonasの中に保存可能？って思ったけど、公開情報じゃないとダメだから難しくね？
                ii. プライベート分散SNSをどう実現させるのかに近い話なのか.....
                    1. いったん思考放棄
        2. 一旦は共有できればいいかな？
            a. 聞きたい
    b. 実現させたい機能
        i. DID認証
            1. 上記でも記載したがこの部分の実装は行いたい
        ii. Key registry機能
        iii. フレンドリスト機能
            1. 設計が複雑になる可能性がある
6. 状態管理機能をどう行うのか
    a. Tablelandのままにするのか
        i. ここは検討する必要がある-> 絶対にもっといい最適方法がありそう

機能整理

- 機能:
    - login画面
        - サインアップ
            - `root_node`と`root_key`の作成を行う
        - サインイン
            - ブロックチェーンアドレスを使用して認証する
            - 署名の検証が成功するとKMS内にある`root_key`の取得を行う
    - My-box画面
        - フォルダの作成
            - メタデータが生成する
            - 暗号化を行いIPFS上に保存する
            - root_nodeにCIDを保存する
        - ファイルのアップロード
            - 暗号化を行いIPFSに保存する
            - 生成されたCIDをフォルダのメタデータに保存する
            - フォルダを再度暗号化しIPFS上に保存する
        - 再暗号化
            - 特定のフォルダorファイルを選択し、最下層に向けて復号する
            - 最下層から特定ノードまで暗号化とIPFS上に保存するを繰り返す
        - 共有
            - 共有相手のアドレスを入力すると共有相手に`file_info`を送信する
    - Shared-box画面
        - itemの取得

                - CIDと鍵を入力することで共有itemを取得する
            - itemの表示
                - 受け取ったitem一覧が表示される
    - Get-box画面
        - 受信
        - 共有された`file_info`の一覧が表示される

メモ:

- 長々と書いているけど、一番はサーバー処理なのか、ブラウザ処理なのかが決まれば次の動きがしやすい
    - ここがまず一番決めたいこと
- で、僕はDIDの部分の設計とリサーチには入っている
    - この部分は実装したい