Story UFC-280: Onboarding

The following document will go over the design of UFC-280, as outlined in Story ticket UFC-280.

Acceptance Criteria

For reference, the following are the acceptance criteria for this ticket:

- After signing up for a user account, I want to see the onboarding process.
- I want to see my current progress through the onboarding process.
- Each step of the onboarding process should be for picking/creating accounts for each account type (i.e. asset, liability, income, expense).
- At each step, there should be a pre-configured set of accounts that I can choose to use.
- At each step, I can add other accounts to make up for those that aren't in the list.
- For each account I create/choose, I can specify an opening balance for it.
- At the last step, I want to see a summary of all the accounts that will be created for me.
- Once I am satisfied with the accounts, I can go to the app and have all the accounts created.
- I want to be able to skip the onboarding process to get straight into the app.

Design Brainstorming

In terms of state management, I guess the onboarding process is quite similar to the import process. As such, its state should be managed by a Redux slice.

Redux Slice

Here's the state we're gonna need:

```
currentStep: number

accounts: {
    asset: {
        [id]: account
    },
    ...
}
```

```
selectedAccounts: {
    [id]: boolean
}
```

OK, so I'm kind of in a dilemma right now. Currently, I've implemented the `SelectableAccountsList` to use the `useSelectableList` context for the selected state.

However, I *think* we need to store the selected state in Redux so that the UI is always in sync with the store data.

If we instead did it so that only the account data gets stored in Redux (such that accounts and selected state is store ephemerally and persisted to redux on 'Next'), then we'd lose the selected state when navigating backwards.

As such, all the data needs to be stored in Redux.

How to Trigger Onboarding

We basically need to decide/design how we're gonna handle getting the user into the onboarding process and preventing them from accessing it again.

Currently, while implementing the layout, I've just had the Onboarding scene on a route (`/app/setup`), but I don't think that's the best way to go about it.

I think that showing the onboarding to the user should be controlled by a database-stored flag, so that during signup/login we can just check for the flag to control whether or not to send the user to the onboarding process. This also means that the user has no way of manually navigating to the process.

Component Breakdown

Atoms

- ListItemCheckbox

Molecules

- SelectableAccountsListItem (ListItemCheckbox)

Organisms

- ProgressStepper (extract from TransactionsImport and refactor)
- SelectableAccountsList (SelectableAccountsListItem)

Scenes

- Onboarding (app introduction + 5 steps)

Tasks

- Build the onboarding process.