Bug UFC-398: Recurring Transactions don't Resolve Historically

The following document will go over the design of the fix needed for recurring transactions not resolving if you miss the day they resolve, as outlined in Bug ticket UFC-398.

Bug

Problem:

Example: If a recurring transaction is set for the first of the month, but you don't check uFincs on the first of the month, then those transactions aren't resolved into real translations.

Also, there's a secondary bug where transactions will re-resolve if you change the amounts and then login to a second device. I guess the duplication checking logic isn't great. Only check against description?

Solution:

??? I thought I had accounted for this use case. Need to dig into the code.

Reproduction Steps

i. Create a recurring transaction that repeats every day, starting today.

ii. Change system date to a week in the future.

iii. Notice that only one transaction is realized: the one a week into the future, none inbetween.

Fix Brainstorming

OK, well I'm writing this doc more-or-less in retrospective. I *thought* this was gonna be a tough(er) bug to crack. I *thought* I'd have to do something like add a `lastRealizedDate` to the Recurring Transaction.

But no, I'm just an idiot. We can trivially derive the 'last realized date' by just *looking at the transactions that have already been realized*. You know, the one *we already create*.

FFS DEVIN! You literally already relate transactions to recurring transactions by a `recurringTransactionId`, and literally have a `selectTransactionsByRecurringTransactionId` selector. Just take the last realized one and use that as the starting point for realizing new ones.

Honestly, I have no idea what made me use `today` as the base for realization. I'm just fucking stupid I guess.

Another Bug: Deletion

Date: January 16, 2022

After another email from a user, I realized that changing the realization algo to "last realized transaction" rather than "today" means that deleting the last realized transaction is impossible. If you delete it, then the algo will just fall back to the previously realized transaction, causing the deleted transaction to be re-generated!

Which means that to properly fix this whole thing, I *do* have to do the thing I didn't want to do but now realize that I have to do (and which I don't think I documented here): store a `lastRealizedDate` property on the recurring transaction itself.

If we have that date and change the algo to use it, then users can delete realized transactions all they want.

So I *guess* it's just a simple schema change and frontend change... might as well do it.

Opened this as UFC-404.

Tasks

- Apply the above fix.

Postmortem

The "Fix Brainstorming" is effectively the postmortem.