REMOTE PANTRY // NORA HAMMACK // MARCH 2018

Description: I want to know what food I have on hand while I'm not at home. Being able to access my pantry remotely will give me more free time, I can meal plan on the fly instead of having to block off time on the weekend to strategize and shop, and it will reduce my food waste by helping me manage leftovers or perishable foods. When I go to the store, I can check my pantry before I mistakenly buy duplicate items, and with the shopping list feature I won't forget to refill basics.

- A user can add items to their pantry or directly to a shopping list
- A user can browse their pantry
- Users can manage pantry items: remove from pantry or add to shopping list
- A user can pull the shopping list they've compiled and mark items as refilled which will move them back into pantry

MVP:

- User management (users have profiles, app can handle multiple users)
- User can add items to pantry ('add' page, pantry data table)
- User can query pantry ('pantry' page, table displays)
- User can modify items in pantry (remove from pantry, add to shopping list)
- @Store mode (pull shopping list, 'store' page, move item back into pantry)

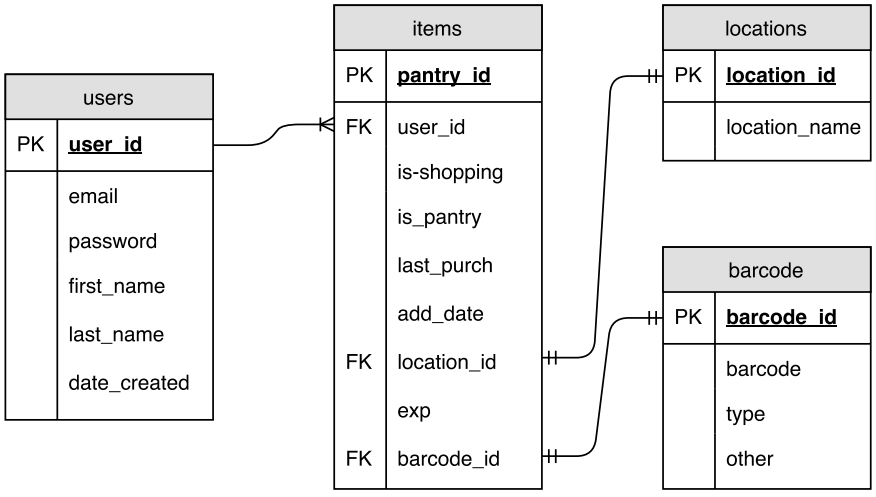
Next steps:

- Bootstrap it!
- Mobile compatibility
- Eat me mode ('eat me' page, filter by expiry)
- Inactive user management
- Semi-preloaded common items (ie 'milk' autofills 'fridge' and 10 day expiry)

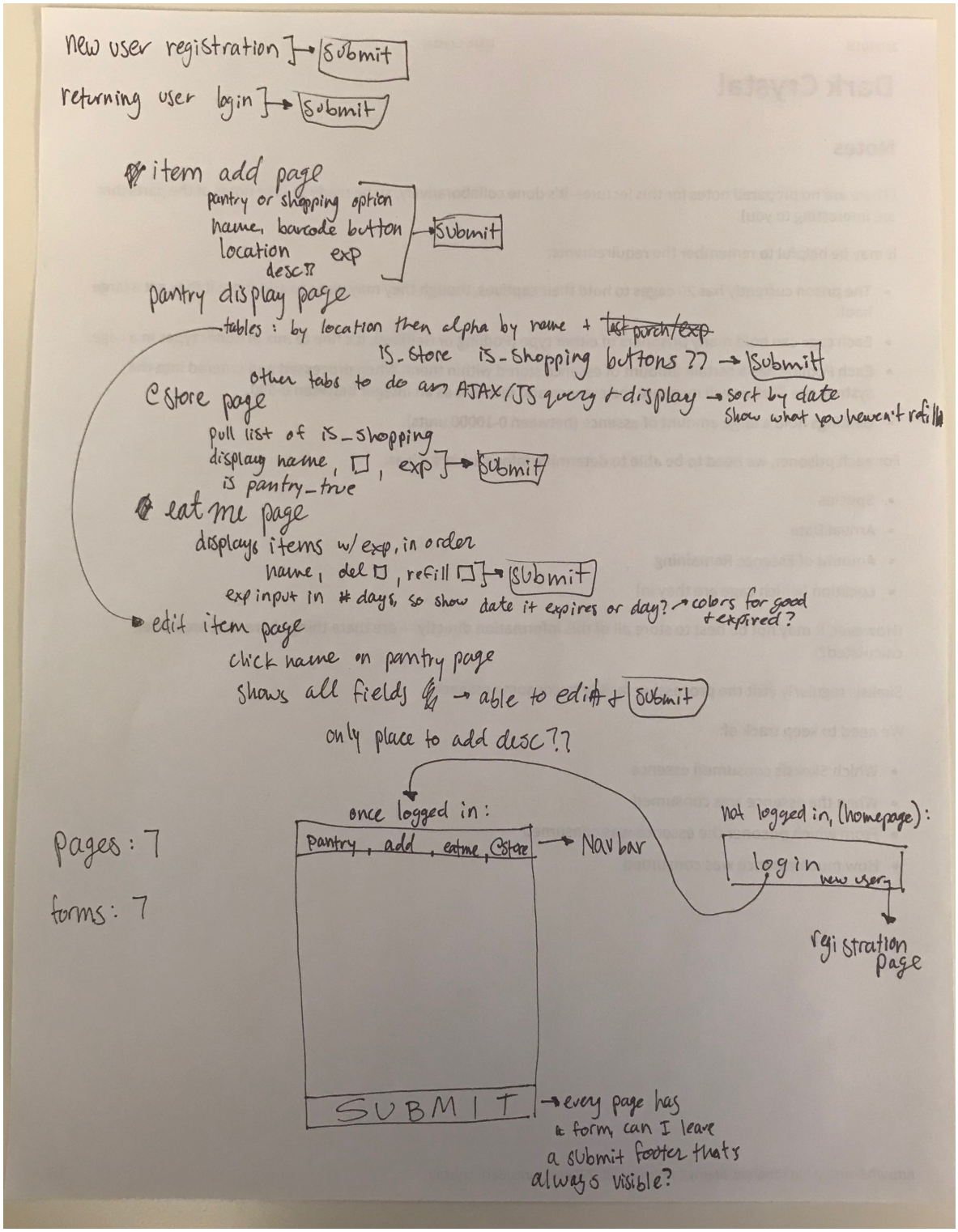
Reach goals/further thought:

- How does my app handle/represent duplicate items?
- Barcode reader: API integration (managed through barcode table)
- Google Home compatibility: Google Assistant or AI API integration
- Spoonacular API integration (find recipes by ingredients)

Data structure:



Routes, page structure:



Order of operations:

- Make tables and database (SQLAlchemy)
- Make pages/routes (Flask)
- Handle forms and users