GitHub Username: SkippyJonney

Prep-Cook Inventory

Description

Prep-Cook Inventory is a quick and lightweight inventory management system that allows you to briskly keep stock of inventory from categorized components.

Intended User

This app is intended for kitchen administrators and cooks who need a lightweight solution to inventory management.

Features

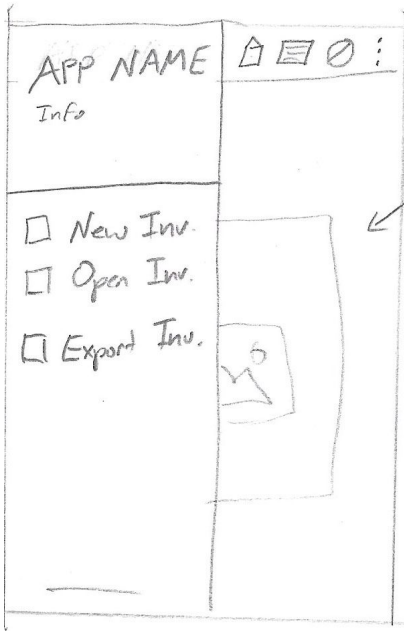
Inventory Features

- Fully implemented CRUD inventory system
- Ability to add fields to inventory items
- Quick query of inventory by name

- Ability to add and filter by “Location / Category” to divide inventory
- Inventory listing with widget to inc/dec order amount
- Export of current inventory in csv format
- Export of current inventory as email

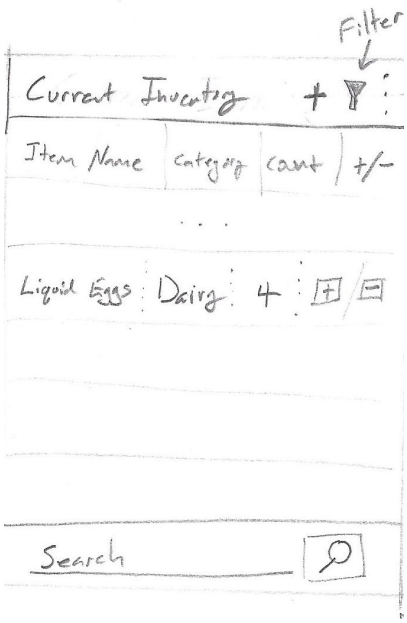
User Interface Mocks

Screen 1

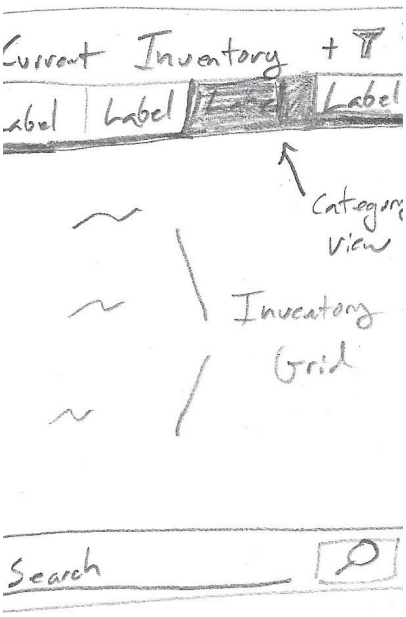


Starting screen has a background image / or blank. Side drawer starts in the open position prompting the user to start a new inventory, open an existing, or export.

Screen 2

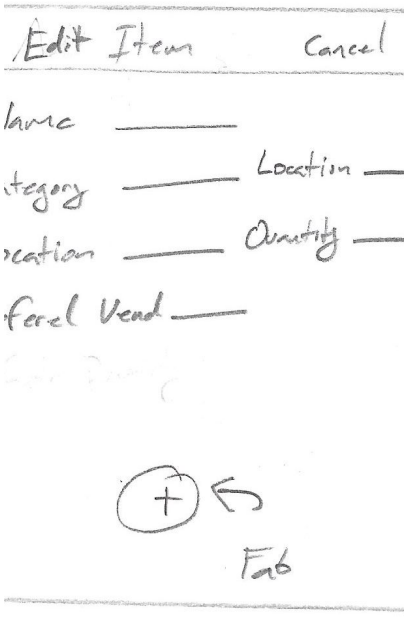


Screen 3

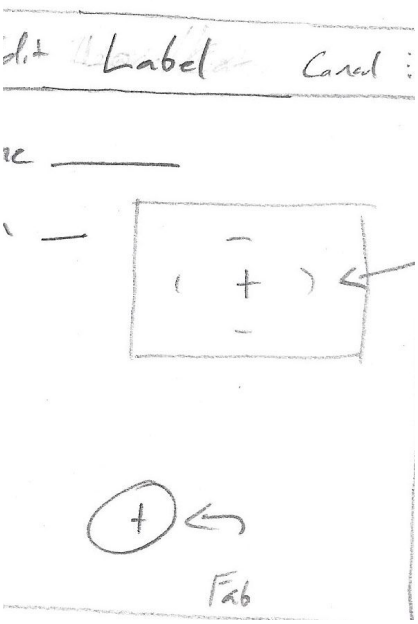


Screen 2 in the main inventory screen. This allows you to search for inventory items and edit their quantity on the fly. A recycler-view provides the list items. Screen 3 shows the inventory screen with filters selected. A paged view is implemented that allows you to only view items that meet a filter criteria.

Screen 4



Screen 5



Screen 4 & 5 show the add item and add label options. The fields are uploaded to the sqlite database when the fab is pressed.

Key Considerations

How will your app handle data persistence?

This app will use a content provider to implement access to an sqlite database storing inventory .

Describe any edge or corner cases in the UX.

If the inventory entry process is interrupted while inside a drilldown menu category, the app will reload at the location.
When using the app on a tablet there will be a master / detail view of inventory categories and items.

Describe any libraries you'll be using and share your reasoning for including them.

I will be using up to date libraries of:

Android Studio v3.1.2
Gradle 4.4
FastCSV v1.0.2 - for csv export support
Android Support Libraries v27.1.1 - for app compatibility and material design
Firebase-core:16.0.3 - for analytics
Play-Service-Ads:15.0.1 - for advertisements

Describe how you will implement Google Play Services or other external services.

Google analytics will be used to see how users navigate a large dataset. Google admob will be used to display an advertisement on some pages.

Next Steps: Required Tasks

Task 1: Project Setup

- Get current version of libraries.
- Configure build.gradle

Task 2: Implement Sqlite Database

- Declare database helpers and contract
- Declare content provider & adapters

Task 3: Implement UI for Inventory Screen

- Create fragment for inventory item
- Hook up recycler-view to content provider
- Implement Loader
- Build drawer fragment
- Add filter ability
- Hook up search with async task

Task 4: Implement UI for Add Item

- Create layout fragment
- Create intent to add item to database

Task 5: Implement UI for Add Label

- Create layout fragment
- Create intent to add item to database

Task 6: Implement Google Play Services

- Add analytics to inventory screen
- Add interstitial advertisement to add item intent
- Test admob

Task 7: Implement UI for Add Item

- Create layout fragment
- Create intent to add item to database

Task 8: Build

- Clean repository
- Add installRelease gradle task
- Add signing configuration and keystore