**Mobile Computing - AndroidOS Final Project Proposal**

**Team Members: Nick Pierce, Matthew Berry, Jared Corder**

**App Name: Smart Shopper**

**Overview:**

One of the most arduous, and dragging, aspects about shopping is locating where all the items are. New customers are always staring at undescriptive signs and asking themselves, “why is this item not here?”. Our team’s app, **Smart Shopper**, is designed to facilitate that predicament, and allow store managers to have the flexibility of moving item locations without all the hassle. This app is comprised of 6 tabs.

Tab Zero (Welcome Screen tab): Initial view displayed to both consumers and admin. Users will select which role and corresponding store and be “intent” over to the appropriate subsequent view. A spinner exist to facilitate the selection.

Tab One (Store Screen tab): A virtual, noninteractive, display of the stores schematic. Inside the display, aisles will be individually labeled with their corresponding department name. Non-aisle departments- such as produce, meat, dairy, etc.- will simply retain a label of its name under its virtual location. Below will be a fixed, navigation bar to permit ease in traversing to other tabs in the app; the home tab on the navigation bar is this tab.

Tab Two (Search tab): When opened, only a title “Search item” and a search bar will initially appear. Intuitively, users will subsequently enter the item’s name in the search bar (e.g.: “Cheez-its”). The search bar retains an auto-complete feature to predict customer queries. Partial queries (“Cheez” instead of “Cheez-it” are accepted); best matches will be determined and a subset will display on tab one- showing all locations of matches and displaying the aisle number and/or department section- varying on if the department retains aisles or not. A fixed navigation bar and a search again button exist on the bottom.

Tab Three (Admin Login): When opened, the app’s logo, username, and password fields are displayed. The admin is prompted to enter a username and password to login into the admin features tab.

Tab Five (Admin Hub): When transferred from Admin Login tab, users are given there employee information. Two buttons will be displayed, each linking to their functionality. One button will go to the create an admin (this is only visible to store admins). The other button will go to product modifier

Tab 6 (Product modifier): Once delivered to this tab from the Admin Login tab, the admin can alter three things: the location of an item, the appending of a new item, and the removal of an existing item. These features are encapsulated in the first three buttons: add, remove, modify. The user will put the barcode in and click one of three buttons: Add, Update or Remove. Clicking all three of these buttons will display the following fields below: Store Name, Vendor, Department, Isle, Search tags. If it is an update or delete action there will be a check to see if that barcode exists, and alert the user if it does not. For update, the fields will be editable and the user can edit the fields accordingly (this may change once the database is created and we establish foriegn keys). If it is a delete the fields are not editable. The user is just seeing them to confirm this is what they want to delete.

If it is an add there will be a check to make sure that the barcode is not in the database. If the check is valid, it will allow the fields to be edited.

Once the user is done, they click the submit button. A status will appear if the submit works successfully. Users can return to hub with the back to hub button

Tab seven(Admin modify): This field allows store admins to create admins.the admin can alter three things: the update an admin, creating a new admin, and the removal of an existing admin. These features are encapsulated in the first three buttons: add, remove, modify. The user will put the admin id in and click one of three buttons: Add, Update or Remove. Clicking all three of these buttons will display the following fields below: Name, Employee id (not editable), Password, and a rank checkbox that indicates if that admin has rank perms,an update or delete action there will be a check to see if that admin id exists, and alert the user if it does not. For update, the fields will be editable and the user can edit the fields accordingly (this may change once the database is created and we establish foriegn keys). If it is a delete the fields are not editable. The user is just seeing them to confirm this is what they want to delete.

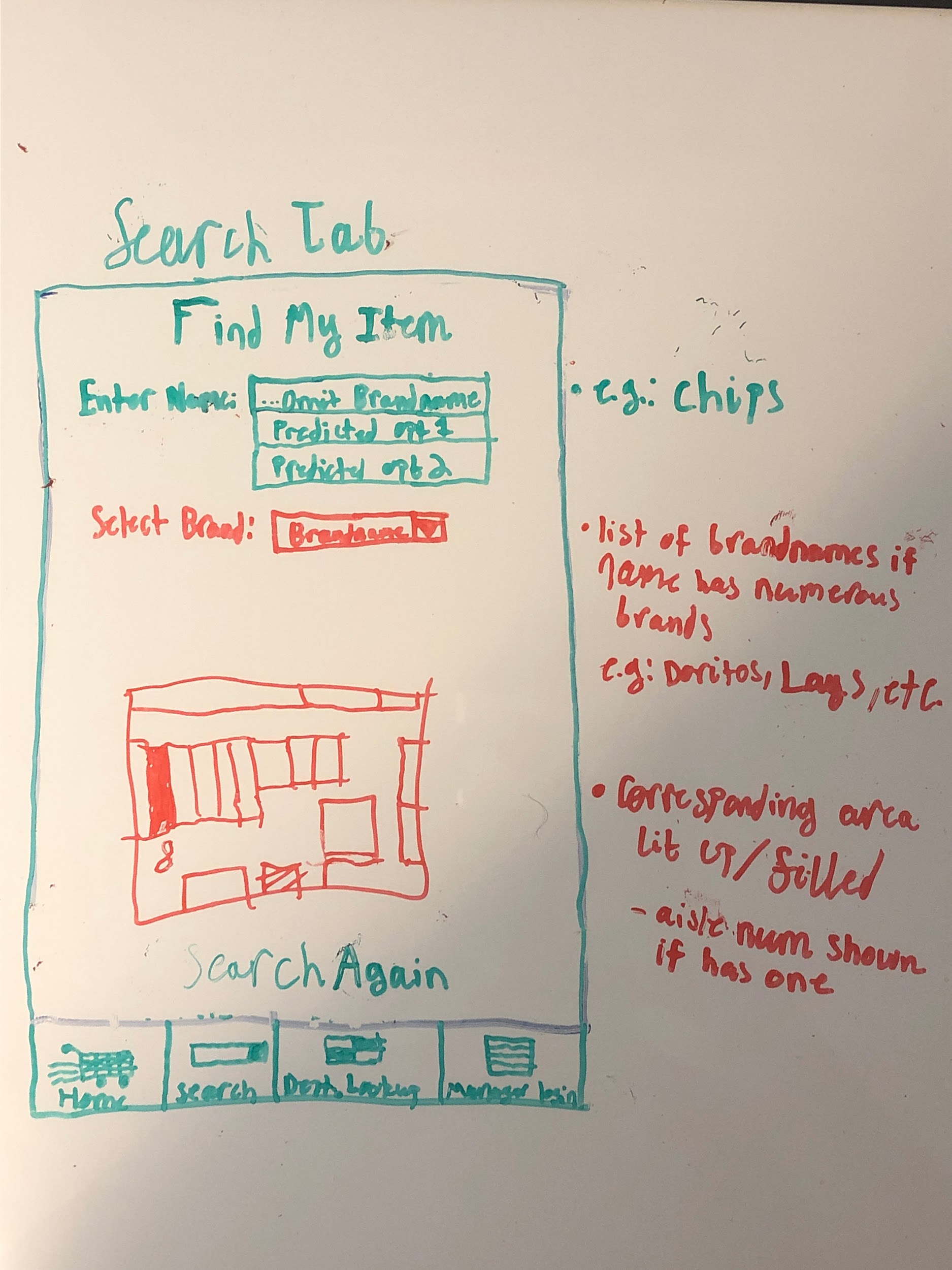
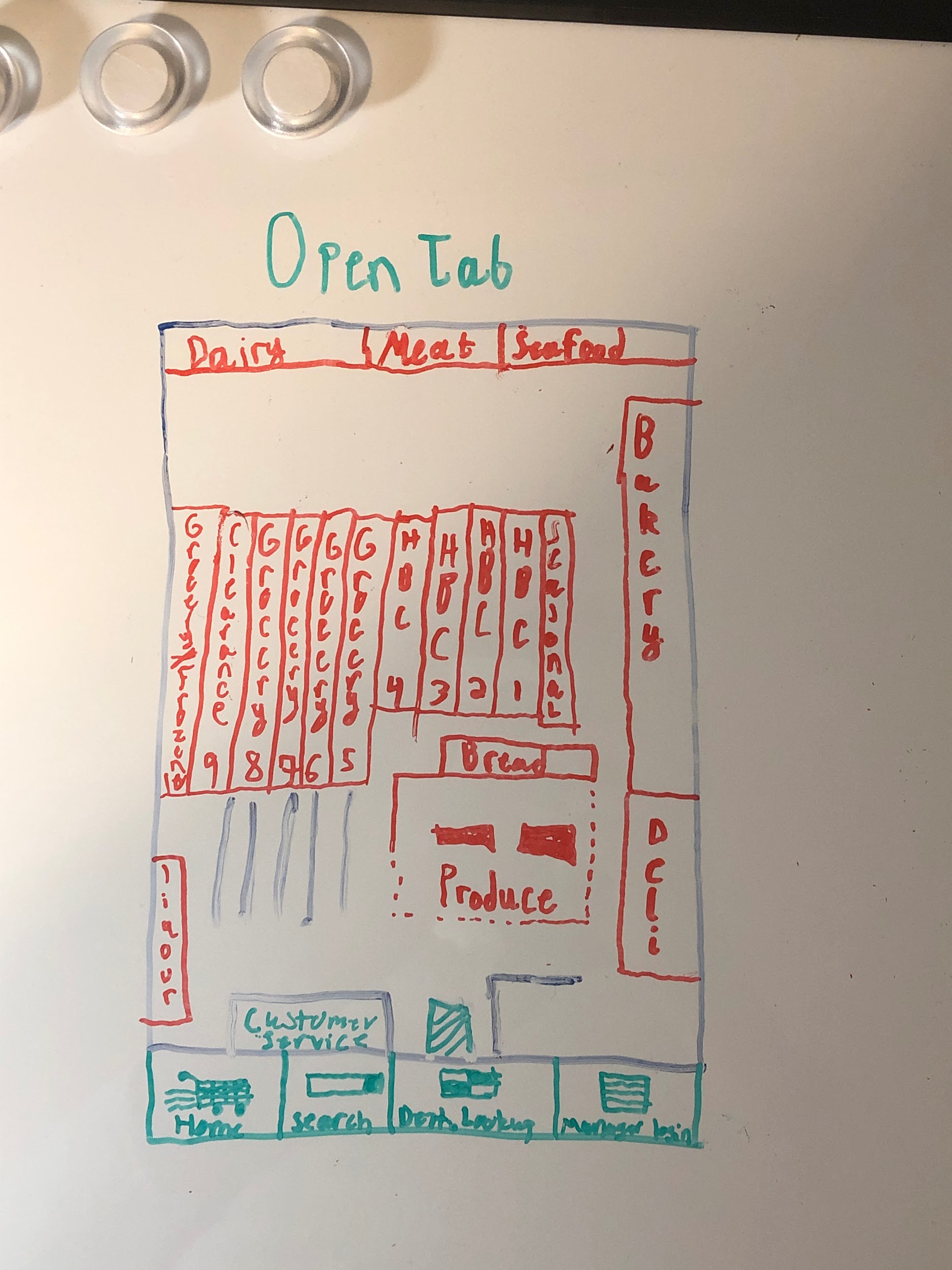
If it is an add there will be a check to make sure that the admin id is not in the database. If the check is valid, it will allow the fields to be edited.

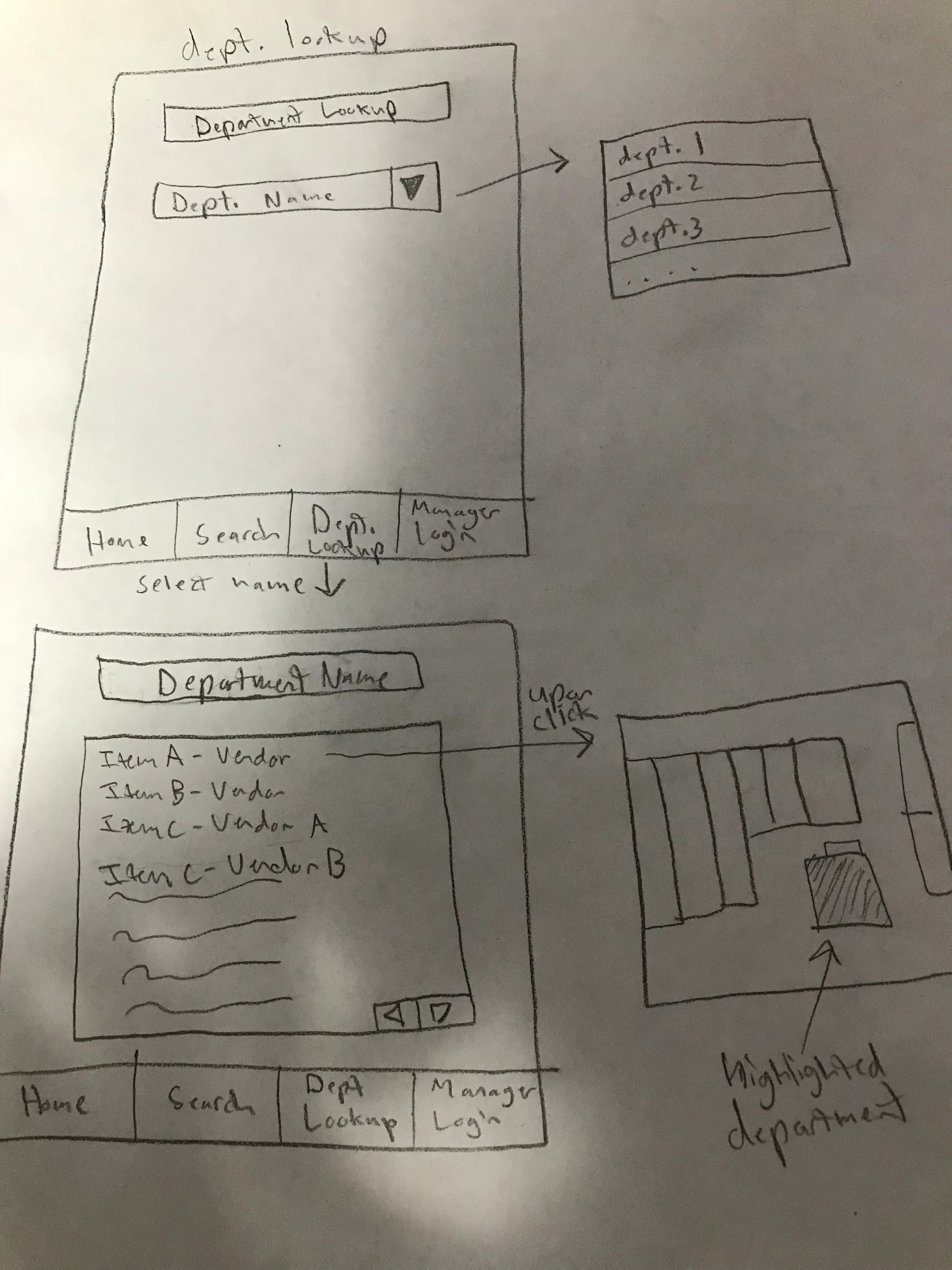
Once the user is done, they click the submit button. A status will appear if the submit works successfully. Users can return to hub with the back to hub button

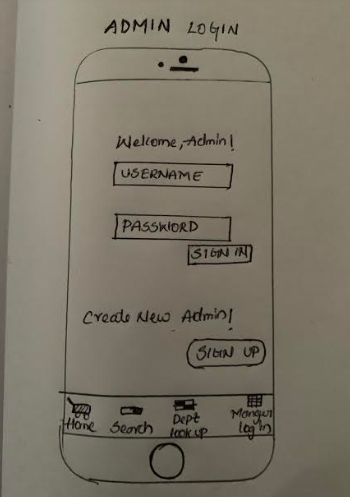
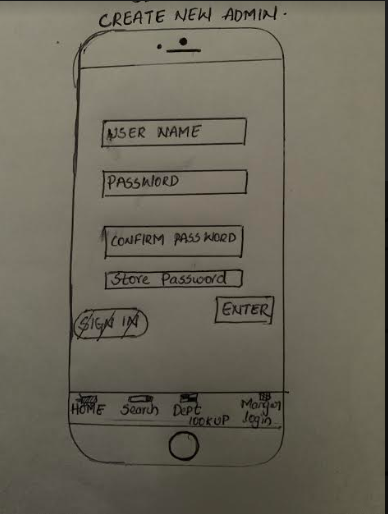
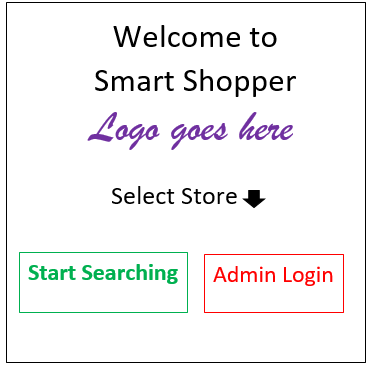
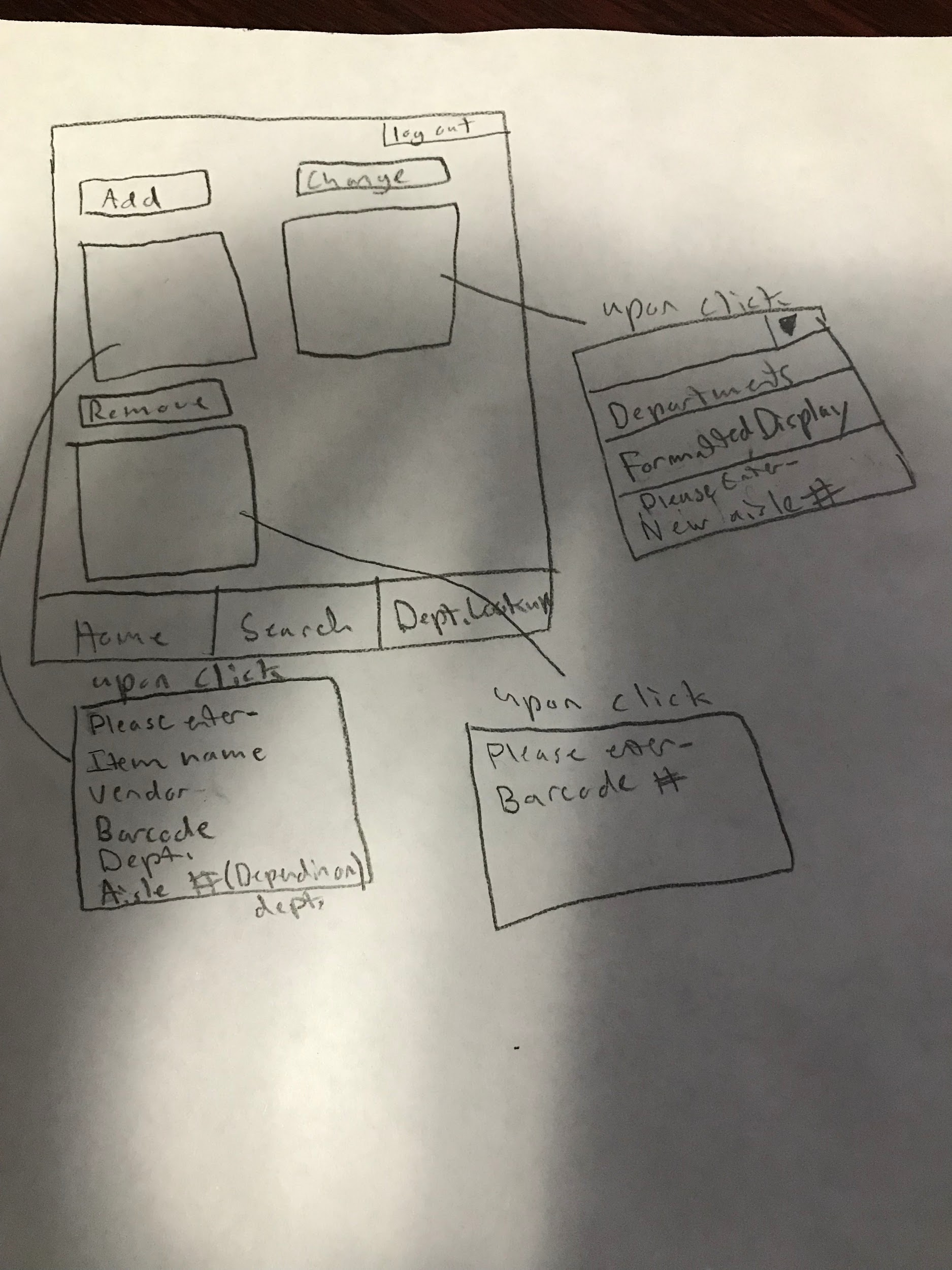
**Risky/External Components:**

1. Utilizes adapter/DLL from MBAAS (remote data persistence)
2. Invokes architectural design and patterns
3. May utilize responsive design to be optimal across all sizes

**Sketches:**

****



****

**Flow:**

Initial activity: welcome screen (store selection).

* Can use nav bar on bottom to navigate to any layout aforementioned above on user side only after initial intent from starting activity. If traversed to login for admin, can enter valid login to proceed to admin activity layouts w/o nav bar
  + If opted into to render new admin, transferred to new admin + store authentication and item layouts
  + If the admin clicks on the logout button at any time then they will be transferred back to the home screen
* If search action is invoked on user side then they are conditionally taken to the the store map screen where they can start search
  + When search is created a text view appears to enter the desired product.
  + That item is then searched and result is determined. **The location of said product will light up, with other locations becoming transparent, to indicate visually where the item is at along with text view w/ auxiliary information.**

Android App: SmartShopper

**Contents:**

- Abstract/ Business Requirement Definition Executive Summary

- Business Requirement Definition

o Consumer Definitions

§ Consumer Expectations & Security

§ Consumer Scopes

o Business Preconditions

§ Project Assumptions/Constraints

- Functionality Requirement Definition

o User Functionality

o User Functionality Business Context/Rules

o App Description

§ Organization

§ User journeys within organization

§ Adapter enumeration and purpose

- System Description/Project Description

o Project Abstraction

o Data Definition

- Team and Roles/ Divisions of Work Agreement

- Project/Deliverable/s Timeline

**Important Note:** This document is not inclusive and many granularities are omitted from the scopes of this project.

**Business Requirement Definition (BRD)**

**Business Need (Brief):**

As grocery stores expand and demand change, a store’s state/content will indubitably fluctuate subsequently. A need exist to institute a paradigm/model that’ll permit users to easily navigate through any store now and tomorrow. This business application servers as an intuitive, water-down proposition to meet those business challenges.

**Abstract/Executive Summary:**

SmartShopper is an application to facilitate the process of maintaining a store by providing intuitive interactions to a variety of possible applications. SmartShopper serves to assist both customers and various levels of admin. Customers can search, and be shown, within a graphical display of the store’s schematic where the item is located. An admin self-service portal exist to permit admin, varying on their level/associated privileges, to perform an inclusive selection of functionality. These include adding, modifying, removing items and admin- respective to their store.

**Project Brief:**

- An Android application deliverable that provides thorough separation between consumers and admin

- Easy to use/understand functionality that permits item look ups and state modification/creation

- External persistence and encapsulated interactions to promote an evolving/ever-changing environment

- Devised, tested, and outputted within targeted due date (12/7)

o Milestone/ SDLC “client” reviews at enumerated dates

§ 10/25

§ 11/15

§ 12/3 (informal standup)

**Consumer Definition:**

Abstract: Two types of consumers are expected to interact within this application: customers and store admin. The functionality, data, and security requested for each consumer is widely disparate.

Customer:

o Expectations: To easily identify, and select, which store they are currently at. To see, and quickly comprehend, the store’s schematic and relative positioning of departments. Paramount feature for consumers will be to look up items by non-technical (see below) names and subsequently be delivered back to the store’s schematic- clearly indicating the item’s specific position. Consumers are expected to search many times.

o Security: No consumer data is recorded between users and interactions.

o Scopes: Consumer features are rudimentary and may be expanded throughout duration of project.

Admin:

o Expectations: To easily identify, and select which store they are currently at. Admins are then expected to login and be presented with the ability to create, modify, and remove store items. Varying on privilege level admin will be able to create new admin, and associate store privileges.

§ Privilege levels and definition:

· Store admin:

o Typical admin. Can freely manipulate item/commodity state (add, modify, remove)

· Managing Store admin:

o Same as prior admin, but with extra functionality pertaining to creating admin. May not remove admin. Only admin they can access/render are store admin.

· Owner:

o Only one per store. Can freely create and remove admin. A system admin to the application must, via remote data store, remove/modify an owner.

o Security/Scopes: Admin state shall not persist locally between sessions. No state at any point of their session shall be openly disclosed.

1. Non-Technical: Categorical names generally associated with an item ie: Doritos Cool Ranch Chip -> Chips.

**Business Preconditions**

**Project Assumptions/Constraints:**

- External extensibility for data persistence will be abstracted, but not separated into a dynamic link library.

o Back4App will be employed as external persistence

- Admin sessions are not recorded and vetted.

o No transactional rewind to any admin modifications will be implemented

- No recoverability of user state/actions will be recorded or rolled back on external complication events.

- App will persist local state/ preclude crashed resulting from activity destruction, but corresponding view design/ layout will reflect viewport’s dimensions in a layout display.

**Functional Requirement Definition (FRD)**

**User Functionality:**

- Customer & all Admin: Identify and select desired store

- Customer & all Admin: Display validation/execution errors

- Customer: Search for specific or general items

- Customer: Receive persistence state bundle, validate, and display resulting location/s

- Customer: Transition to any tab within one jump in customer-app scope

- Admin: Login against remote data persistence

- Admin: Display user credentials and available store-modification options

- Admin: Add items by passing unique identifiers and other needed commodity state in an intuitive fashion

- Admin: Remove items by passing unique identifiers

- Admin: Modify items by passing unique identifiers, on acquiring the item, admin may change non-categorical state/pliable state of said item (see below)

- Managing Store Admin: Create and remove new Store Admin

- Owner: Create new Store or Managing Store Admin. Remove Store admin via drop box display

*Non-Categorical: state/ attributes of an item that is not considered a primary key nor index to a relation/s*

**User Functionality Business Context/Rules:**

Important: rules enumerated below are general and are subjected to modification per business need clarification.

- Customer Search for specific or general item

o A user search phrase must be longer than 3 characters

o Search results from user query cannot exceed 4 total

o If multiple departments exist within item list- ie: 3 items returned with 1 being in a different department- then the user should be able to select which department the item/s they want

§ This is done due to that a simplified search, such as “chee”, will yield both “cheez-its”-grocery and “cheese”-dairy

o If multiple items and departments exist within search set, then customer asked to rephrase search phrase.

- Customer: Receive persistence state bundle, validate, and display resulting location/s

o Results should be saved and not refreshed in an on destroy event

o Schematic should “light up” corresponding area where results preside in

o A “search again” button should appear allowing users to go back to search- this will trigger a refresh on schematic display, meaning that all colored location indicators will light up again

- Admin: Display user credentials and available store-modification options

o Store-modification options shall only appear/ be available to the admin if the admin retains the privileges to do so

- Admin: Add items by passing unique identifiers and other needed commodity state in an intuitive fashion

o Commodities’ id must be unique (barcode number)

o The item’s name shall be unique to the vendor

§ If the vendor name doesn’t exist, create it

o The item must have a categorical name

o The item can only pertain to one department

§ If the department has aisles, then the item must have an aisle within the given range

o Price required

§ Non-negative

o Supply required

§ Non-negative

- Modify items by passing unique identifiers, on acquiring the item, admin may change non-categorical state/pliable state of said item

o Any non-categorical state that is changed must be revalidated to the conditions of adding the item ie: price non-negative or the name-vendor matching is unique

- Managing Store Admin: Create and remove new Store Admin

o Can only create or remove store admin

o Create- Admin username unique

o Create- Admin password 8-characters

o Remove- Admin username is passed

- Owner Create and remove any admin

o Can only create or remove store admin

o Create- Admin username unique

o Create- Admin password 8-characters

o Remove- Admin username is passed

**App Description**

**Organization**

SmartShopper Android is comprised of 7 views: Welcome, StoreSchematic, Search, AdminLogin, AdminSelection, AdminItem, AdminCreateAndRemove. Varying on the user selection- what type of user they are- only 3 to 4 views will be utilized. A customer will be brought to the corresponding store’s schematic and can iteratively search for items (see user functionality). Admin will be asked to login, and can perform a variety of functions varying on their privilege level (see user functionality, user preconditions/ business rules, Consumer definition). Tab bars exist to allow customers to switch to any view freely. Admin’s do not retain a tab bar and resort nav-bar navigation.

**User Journeys**

Prelude: user journeys are not exclusive and variations do exist. This section is to indicate the general flow of user interactions and inputs/outputs. Validations in user input/interactions are posited.

· All (partial-all consumers begin here)

o WelcomeScreen

* User ask to select store and self-identify consumer-role.
* Consumer role: transition to store schematic.
* Admin role: transition to admin login

· Customer Store Schematic

o Non-interacting view that display the store’s schematic and indicates the relative positioning of departments

· Customer Search

o Customer enters search phrase (search phrase can be the technical name or the simplified/categorical name). If search results exceed four the user will be ask to append to their search. If no search results yielded user ask to rephrase/retype search phrase. If search results retains more than one department, or less than four items, then users are displayed a recycler view to display the options to the user for specification. On valid search results, or after user specification, a transition to the store schematic will transpire.

* Store Schematic: upon receiving the selected item, its department state will be extracted, and its corresponding department position will be displayed. A “Search Again” button appears allowing for transition back to search tab, clearing/refreshing view and local state.

· Admin Login

o Admin enter username and password- and upon entering a valid login- are transitioned to the Admin Selection view

· Admin Selection

o … Mathew describes how his view/user interactions transpire with respect to business rules enumerated above (see User Functionality & User Functionality Preconditions/ Bus

**Integrated Adapters**

* Back4App
  + service/s: provides remote persistence of stored state with a corresponding DLL to permit a variety of manipulations/accessibility
  + Purpose for grocery store: to remotely store, and modify, state pertaining to SmartShoppers data: store instances, commodity instances, department instances, and admin instances.
    - Data Cardinalities and definitions:
      * Store: exist to represent the store’s name, location, and size
      * Departments: exist within a store to encapsulate a certain category of commodities/groceries. Departments retain their type, if it has aisles, minimum aisle number, and maximum aisle number
        + Departments must belong to one store.
      * Commodities/Groceries: represent individual items that a department within a store possesses. Commodities/groceries retain a barcode, name, vendor
        + Commodities/groceries belong to 1:m departments
        + Price and aisle number are stored in each relation
      * Admin: represent a worker who has obtained admin privileges. An admin holds the name of the employee and his/her’s privilege level.
        + Admin belong to only one store
    - Data Restrictions (only including ones relevant to SmartShopper):
      * Commodities/groceries’ relation to department will have a cascading delete to the commodity relation. Ex: If I delete doritos from my store then all doritos in the associate will be deleted subsequently
      * No restrictions pertaining to stores or departments (can’t be created, updated, or deleted via application)
      * No restrictions pertaining
  + The android application will invoke its DLL to perform the following crud options, with justifications, on valuable business state:
    - Stores: Read for store selection on welcome panel
      * CUD operations are delegated to system admin portal via Back4App
    - Departments: Read for validation of appending items (retrieve vendor from items), updating items (retrieve state from DepartmentStock), deleting items (retrieve item from DepartmentStock)
    - Stores: Read for retrieving departments contained in store
    - Commodities: Read for acquiring vendor from item

**System Description (SD)**

**Data Definitions**

Commodity:

* Barcode: String
* Name: String
* Vendor: String
* Price: Double
* Location: Location-- enum
* Department: Department
* \*\*\* Price, Location, and Department not stored in DepartmentStock

Admin:

* Name: String
* Username: String
* Password: String
* Privilege: AdminLevel-- enum
* Store: Store

Department:

* Type: DepartmentType-- enum
* HasAisles: bool
* MinAisle: int (acquired ordinal value from Location)
* MaxAisle: int (acquire ordinal value from Location)
* Store: Store
* \*\* HasAisles, MaxAisle, MinAisle, Store acquired from StoreDepartments

Store:

* Location: String
* Name: String