**Waterfall SDLC Methodologies –** I believe this will be the best Software Development Life Cycle for us to follow

In the Waterfall model, tasks and phases are completed one by one in a strict order. You need to finish one phase before moving to another one. Plus, there is no going back. And every stage depends on the previous one.

Advantages:

Easy to manage for the team

Simple to explain to the clients

Plan and schedule with clear milestones

Errors are easy to verify at each stage

Technical documentation is easy to understand

Disadvantages:

Inflexibility

High risk of bugs

Compared to the Agile model, it takes more time to make the final delivery

Difficult to measure client needs in terms of a functional specification during the requirement phase

To sum up, this SDLC model is easy to manage a stable and clear definition of the product. But the fact that there is almost no room for revisions once a stage is finished, fixing any problems is quite a challenge.

Design Phases:

Discuss App Capabilities (Initial & Future)

* Initial:
  + **Barcode scanner**
    - **Manual input** if there is no barcode/barcode is not scannable
  + Functionality on **phone or tablet** (should be inherent)
  + **Inventory In/out** (increment/decrement)
    - **Change inventory** by a given number
    - **remove item** from inventory (Delete tuple)
    - **Add item** to inventory (create tuple)
  + Inventory data saved to **database**
    - Attributes:
      * **Item\_ID** (Barcode) *Primary Key*
      * **Item\_Name**
      * **Quantity** (Inventory)
      * **Price** (N/A for non-sales items)
      * **Item\_Type** or **Department** (N/A if not applicable)
  + **Sign-in** for company to manage their **personal database**.
* Future:
  + **Personal sign-in** for employees under the company database.
    - Add Table: **Employee**
      * Attributes:
        + **Employee\_ID** *Primary Key*
        + **Employee\_Name**
        + **Password**
      * **Add employee** (Create tuple)
      * **Remove employee** (delete tuple)
    - Database actions **tracked by employee**
      * **Altered\_By** attribute in database

Select and Build Database:

* Find a host
* Figure out how to connect app to database
* will sign-in be managed through database?
  + test sign-in
* test connection
* test database entry, alteration, retrieval, and storage.

Design App:

* app name
* logo
* include our developer name (shift team 6)
* buttons for all capabilities
  + sign-up (opens window to input info and sign-up but doesn’t do anything for now)
  + log-in (no functionality for now, continue to app main/home)
    - barcode scanner / manual input
      * add new item
      * delete existing
      * sell item (decrement inventory)
      * adjust item info / quantity
    - view inventory
      * adjust item in inventory (may have to include after database connection)
      * delete item (may have to include after database connection)
    - back button (on almost every page)
    - home button (on almost every page)
  + log-out (no functionality, return to log-in page)
* layout
  + where will the buttons be?
  + will there be multiple “pages?”
* Appearance
  + color, shape, and size of windows and buttons
  + font types and sizes
* App should be partially functional
  + all buttons should “work” and lead to next step where applicable.
  + button’s with specific functionalities should print a simple text as a test.

Connecting the Database:

* Connect the app to the database and run some simple tests:
  + input test items
  + display inventory
  + delete test items

Code the App:

* sign up (saves input info to database to retain login ingo
* log-in (must now include functionality. reads existing login info from sign-up.)
  + barcode scanner / manual input (Scanner must access camera and read barcodes to produce an item\_ID. Manual input will simply enter item\_ID or item name. Either will match you with an existing item or let you create a new item)
    - add new item (if item scanned/input is not already in system. write code to create this tuple and save to database)
    - delete existing (write code to delete selected tuple from database
    - sell item (write code to decrement inventory of an item)
    - adjust item info / quantity (write code to change any and all aspects of an item)
  + view inventory
    - adjust item in inventory (reuse code from adjust item info)
    - delete item (reuse code from delete existing)
  + back button (already coded)
  + home button (already coded)
* log-out (log-out the current user, return to log-in page)

Testing and tweaking:

* create multiple log-ins
* log-in to one (to be used as main)
  + create a mock inventory and insert several new items (manual input)
  + log out
* log in to another account
  + check that main account’s inventory is not shown in alt account
  + log out
* log into main
  + check that inventory remains (view inventory)
    - adjust all info of an item in inventory (excluding item\_ID)
    - delete an item
  + manual input an item code
    - sell item
    - adjust item
    - delete item
  + scan a new barcode and input a new item into inventory
    - repeat a few times
  + scan an existing barcode
    - sell item
    - adjust item
    - delete item
    - repeat with several scanned items
  + log out