Functional Requirements:

1. A user can log into the system.
    a. A new user can create an account and log into the system.*
    b. A user can logout of the system.
2. A user can manage their “my medications” list.
    a. A user can add a medication to their “my medications” list.
    b. A user can remove a medication from their “my medications” list.
    c. A user can edit an existing medication in their “my medications” list.
    d. A user may add name, company, price, and quantity metadata to a medication.
        i. A user may alter medication metadata.
3. A user can manage their farms list of groups.*
    a. A user can add a group to their farm.*
        i. A user can define the size of a group in terms of heads.*
    b. A user can remove a group from their farm.*
4. A user can “administer” a medication to a group or “subgroup.”
    a. A user can create a new administering session.
    b. A user can define the group to be administered to.
    c. A user can define the individual tag numbers in that group being administered to.
        i. A user may also select that the entire group is being administered to.
    d. A user can define a product being administered in the session.
        i. A user can define the quantity per head being administered.
        ii. A user can define additional products being administered for the session
5. A user can view history of medication usage.
    a. A user can view the history of all medications administered over time on a per application (not per session) basis.
6. A user can view summary analytics for medication usage.
    a. A user can view the total quantity administered for each product used.
    b. A user can view the total cost associated with each product used.
    c. A user can view the pull rate for each of their farm’s groups.
    d. A user can view which individual tags have had medications administered.
        i. A user can view how much and what has been administered to a specific group / tag number.

* - Can/will be removed after integration with Dustin’s system.

Non-Functional Requirements:

1. Data must remain integrable when user enters it manually.
    a. Must be consistent with the data already populated on the database.
2. Many users and accounts should be able to interact with the system at any given time.
3. Many instances of a single user account should be able to interact with the system at the same time with proper interleaving.
4. Inactivity timeouts
    a. If a user/account has been idle for too long the system will log them out.

5. Must be available with LPA of 99%.
6. Event/action failure will keep the data that has been entered and not erase it.
    a. Retry the action/ request with the same data.
7. No loss of data should ever occur
8. User login and authentication system should use Firebase encryption system.
9. All new user accounts should be setup in less than 3 seconds upon signup.
10. Application should still be functional with limited or no connectivity, and should update the database as necessary when regaining connectivity.

Constraints:

1. Site will only be available through web application.
    a. Must perform on multiple web browsers.
2. Must be modularized in order for Dustin to connect to his database.
    a. Adding and removing features should be as simple as deleting/adding a module.
    b. Conformance to coding standards and best practices.

Concerns:

1. High usability
    a. Basic user can navigation and perform tasks easily.
2. Navigation must be intuitive.
    c. Adding, scrolling, moving, inserting must be consistent throughout application.