**Add this to architecture document**

**CouchbaseLite Class Structure**

Design Principle

Information hiding – This class is implemented to hide details on how to interact with the local database. Specifically, any functions, which need to interact with the database, will utilize this class throughout the application.

The CouchbaseLite class implements all functions that are required to interact with Couchbase Lite database on the device.

**Secrets**: The algorithms used to interact with couchbase Lite API and logic to utilize Couchbase Lite database.

**Services**: Specifically, the class implements create, retrieve, update and destroy functions for Couchbase lite database.

**Error Handling Module**

Design principles

Information hiding – This class is implemented to hide details on how to handle errors. This class is designed for change, because error handling will change based on end user feedback

Single responsibility principal – There should be one central module, which deals with handling all errors. Handling each error individually inside functions could lead to inconsistency and make code maintainability harder.

**Secrets**: How an error should be handled.

**Services**: Specifically, the class implements error handling for each type of exception, which could be thrown throughout the application and the appropriate reaction.

**Add this to Details document**

**Error Handling**

As per the pseudo code, functions throw appropriate exception when an error occurs. There will actually be an error-handling module called in the catch statement, which will handle all of the error in the program. The design will be a simple mapping between error code and how the error will be handled. In some cases for instance, when scanning an invalid barcode, the error is handled by propagating the error to the UI. Similarly, error which result in unexpected behavior will be propagated up to the UI. However, other errors such as replication fail due to network connectivity could be ignored and logged in the application log dump for debugging. They will not be propagated up to the UI. This module will most likely be changed according to feedback from end users.

Code example

try {

//example function

populate\_array(array\_of\_items);

} catch (IndexOutOfBoundsException e) {

/\* Here error handler class decides how to handle the error.

Ignore or propagate error up to UI? \*/

handle\_error(e);

}

**Communication Protocol**

There is only one communication protocol used within server-client inside of our application. This is used to pull information from the couchbase server. However, the application simply calls Couchbase Lite API to pull data from the server. The details are implemented by couchbase Lite class itself. An example of HTTP get is given below to pull data from server using a Mac terminal.

Code example

curl -X GET 192.168.2.12:4984/couchbaseevents/1