Contxt – Project Summary

# Overview

The purpose of this document is to document remaining work, as well as provide details of the import feature. If desired, additional information can be provided regarding the structure of the app (domain model), overall workflow, and how processing works in the background for submitting to the server.

# Remaining Work

## iPhone App Side

There are still a few items remaining that need to be handled before submission:

* Update hostname to be contxtapp.com (or whatever the domain name is)
* Update queries to match path/filename for server
* Very basic testing to ensure the app works (~15 minutes)
  + Any time needed to troubleshoot or for additional testing beyond this will require per hour payment.
* Update Appirater (rating system in app) with contxt app store path (after creation in app store)
* Build for app store distribution

## Web Server Side

From this perspective, there are several things that have already been done, yet several things remaining.

#### Database

The overall DB design has been completed. There may be one or two fields that need to be added since a few Core Data (iPhone app domain model) changes have been made, but it would be something small like adding a field here or there. Overall design is complete. Regarding accessing data, most all Creates and Deletes--with the exception of those for the “UpdateRequests” table, which tracks update requests from the iPhone app and subsequent acknowledgements from the iPhone app--of the CRUD operations are complete in the form of stored routines (procedures) and functions. Some of the Read statements are complete, but those need to be finished, and the Update statements must be written.

In summary for the database:

* Finish Delete statements (as stored routines; this is only DeleteAnnotationDocument and DeleteImageInfo)
* Finish Read statements (as stored routines)
* Create Update statements (as stored routines)
* Add a field called fontSize (short/int) to the DrawingAnnotation table. This field can be null. The associated queries should be updated to incorporate this change as well.
* There may be other differences once server side web code is written to process the requests below, but they should be minor.

Details on *UpdateRequests* table:

The server needs a way to track what is new information to the requesting client app. Whenever the app asks the server for new AnnotationDocument objects or new Annotation objects, etc, the server needs to respond with new information. However, how does the server determine what is new? This can be done by tracking the response that the server provides to the client. The server would then wait for a confirmation that the client processed the response appropriately. When the confirmation is received the server sets a LastUpdated field to the timestamp of the acknowledgement so that future queries only look for new information since that timestamp. This would be the workflow:

* Client requests what is new
* Server checks DB to see when the last time this particular client has asked for updates and successfully processed the response (stored in the UpdateRequests table)
* Server then finds all entries in the DB that are new or updated with timestamps newer than the last time the client update was successful
* All entries are returned in the response to the client
* The client then processes the updates one by and sends an acknowledgement back to the server
* The server then tracks when the last successful acknowledgement occurred in this batch of objects that were returned and uses this as the new basis for determining “what’s new” for the client
* **NOTE: Some modifications to the Database may be required or desired by the developer(s) that implement the server side and DB code. If this is the case, they can do it however they choose, but they must make sure not to send duplicate data (data that has already been processed by the client) and that they do not lose data (don’t track the acknowledgements and assume everything was successful). Either of these could cause unintended consequences in the app or the user experience.**

#### Web Coding

All of this work needs to be completed. This includes, but is not limited to, the following:

* Set up a hosting account for the domain of choice (I think it’s [www.contxtapp.com](http://www.contxtapp.com))
* Create the directory structure that will be used for communication
* Identify what to name data pages to create according to the API documentation (separate document) for iPhone App / Server communication
  + I don’t care what they call the pages, so long as they meet the specs
* Create code in the respective data pages to read input data in the form of JSON (JavaScript Object Notation) and provide the appropriate responses
  + Note: The structure of the input and responses is critical. This needs to exactly match the API specifications. This information will be provided in that document.
  + For clarity, the process here is to read the JSON input, storing the data into temporary DTOs (Data Transfer Objects) and then formulating the appropriate calls to the database.
* Code needs to be written to test their server side logic extensively.
  + This is where we could really waste time later when putting the two together just before launch. If they do a bunch of work and don’t test, and then we put them together and have a bunch of bugs to work out, that is going to be costly.

# Import Feature (“Open in App”)

Import – Sometimes has issues when importing PNG files

Import – Is slow for larger files with no indication that it’s importing

# Submission to App Store

The following items will be required for submission to the app store:

* Company Name
* SKU # (contxt001?)
* Availability Date
* Pricing
* Discount for Educational Institutions?
* App Store Countries to sell in
* Version Number (assuming 1.0)
* App Description (this is what shows up in the app store)
* Primary Category
* Secondary Category (optional)
* Keywords (for search terms)
* Contact Email Address
* Support URL
* App URL (optional)
* Review Notes (optional)
* EULA if you want to provide your own
* App Icon
* App Screenshots or promo images (up to 5)

\*NOTE: As long as this information is provided beforehand, it shouldn’t take more than 30 minutes. Any time spent beyond those 30 minutes will be billed at the hourly rate.