DELIVERABLE #4

University of Toronto Scarborough

Authors: Brandon Chow, Daniel Persaud, Jason Zheng, Kenny Lam, Wilfred Wong

Date : Nov 10, 2015

Table of Contents

1 Current State of Project

Our original plan for sprint#2 was to complete user story id#9 and #3 (notification feature and add custom field feature respectively), however we had to push those to next sprint as we still had unfinished tasks from the previous sprint that had a higher priority. We didn't have much time to do work for the initial 3 days of the sprint as we were busy with other course work (midterms, projects, etc...). During the next week we continued working on unfinished tasks carried over from the first sprint. Our meeting with the TA gave us some insight on the progress we made so far as well as some areas of concern about our implementation. Since we decided to use an API wrapper (pyzotero), we were limited on what we could do with the Zotero database. In addition, we could only get direct access to the online database which means that our application requires the user to have constant internet access to use its features. The application completely bypasses the local database and syncing process. On November 4, we met with the client to discuss our current implementation and any issues we had. The client showed concern when we metioned that our implementation required internet access to take advantage of the new features. The client insisted that she would have to be able to use the application offline since she may not always have an internet connection while working. After the meeting with the client, we immediately held an emergency meeting with the business side to discuss our plans going forward.

At the business meeting we discussed the two options we had: continue using the Python implementation and create our own SQL queries to query the local database, or switch to using JavaScript and use the native Zotero API to query the local database. As this was a huge decision which greatly affects the entire project and plan, we spent some time exploring the two options and weighed the pros and cons of each. If we continued with the Python implementation, we would have to take into consideration all the sanitization that Zotero does with their queries, as well as creating a function to update the entire database (which was around 1200 lines of code). After a long meeting with business, we decided that the best course of action was to switch to the JavaScript API and create a plugin. This gave us the freedom to use the local database however we liked, but this also meant that we would have to re-write a large portion of the code that was already written in Python. The UI would also have to change from using .py files to using .xul files.

We also re-evaluated/updated our estimated costs, values, and risks during our meeting with the business. Our estimated costs/risks have gone up a little since we initially did not expect to make such a drastic change. The estimated cost for user story id#2 and #3 have each gone up by 10 hours, and the risk level of those 2 user stories have increased by 1 (ie, if it was medium, it is now high).

****Note: There were no changes done to the user story content/personas.**
*****Note: The system diagram has changed to reflect the language change to JavaScript (see "System Design Diagram" section)**

2 Product Backlog

Product Backlog				
User Story ID	**User Story**	**Value**	**Risk**	**Estimated Cost**
1	As Anne a professor, I want to be able to batch select tags so that I can perform deletes/renames/and merges on all of them at once.	3	~~Med~~ High	~~20~~ 30 Dev Hours
2	As Anne a professor, I want the option to couple documents so changes made to one instance of a record will be reflected in the copies.	4	~~Med~~ High	~~20~~ 30 Dev Hours
3	As Britney a student, I want to be able to add more fields to an existing record type so that I can classify the record better.	4	Med	40 Dev Hours
4	As Anne a professor, I want an easy-to-use UI for creating custom bibliography styles so that I don't have to fiddle with XML files.	4	High	~~20~~ 30 Dev Hours

5	As Britney a student, I want the ability to work offline so I can work while commuting.	1	High	100 Dev Hours
6	As Anne a professor, I want to have a log of the changes I've made so that I can keep track of my progress.	2	Med	30 Dev Hours
7	As Britney a student, I want to be able to export custom bibliography template styles so that I can share it with others.	2	Low	40 Dev Hours
8	As Britney a student, I wish to import my records from other bibliographic tools that I have previously used.	1	High	30 Dev Hours
9	As Anne a professor, I want a notification feature so that I can come back to records that are incomplete.	3	Med	30 Dev Hours
10	As Anne a professor, I'd liked to be able to attach a photo to my records so that I can visually identify a record by the book cover.	1	Low	20 Dev Hours

11	As Britney a student, I would like a feature to automatically create and update a local backup of my records so that I don’t need to worry about losing my work.	3	Low	10 Dev Hours
12	As Anne a professor, I’d like to have email integration so that I can share my work with my colleagues without opening another email client.	1	Med	30 Dev Hours

3 Sprint #2

3.1 Sprint Backlog

Our sprint backlog is located here:

- User Stories/Tasks that were swapped out are colored in red
- User Stories/Tasks that were swapped in are colored in green

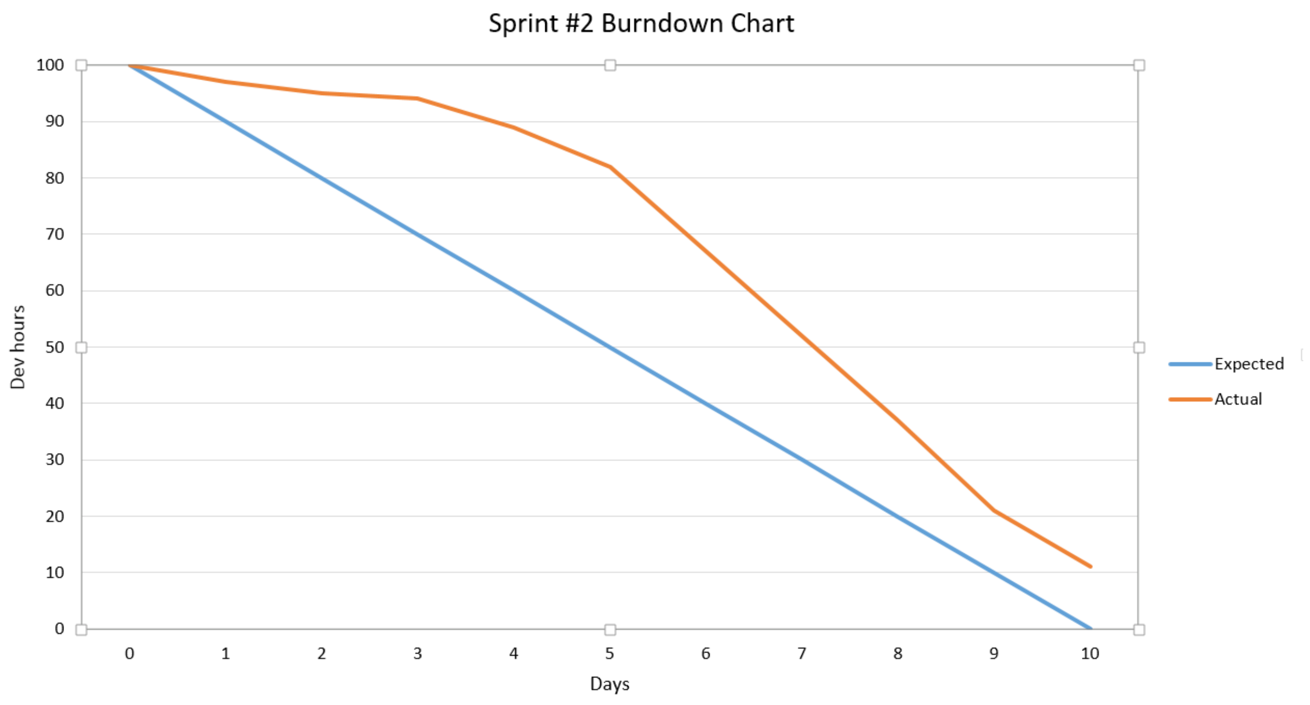
https://docs.google.com/spreadsheets/d/1x4ZO0lsaXktc8lxdIhvAAT5QEFSecOz8eK0PieLRWok/edit?pli=1#gid=1707570274

3.2 Sprint Task Board/Plan

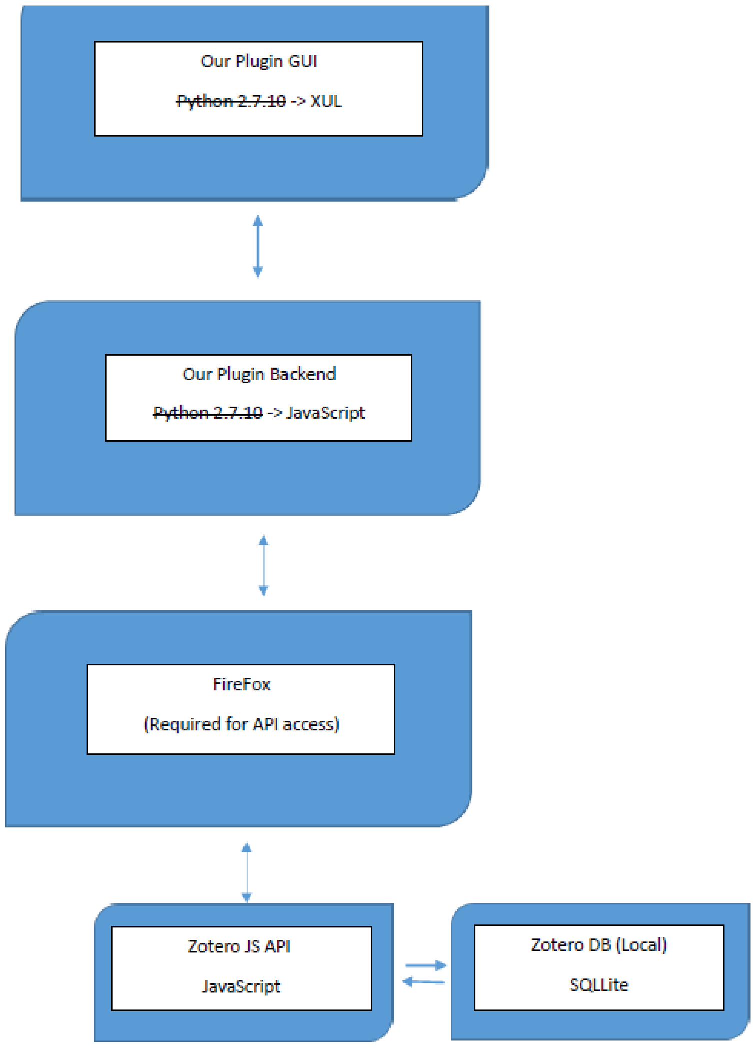
Our task board is located here (snapshots are located under "INSTRUCTIONS"):

https://trello.com/b/xbNi70ao/sprint-2-task-board-precision-development

3.3 Sprint Burndown Chart



4 System Design Diagram



Reason for Changes:

After meeting with the client priorities for certain functionalities of the project have changed. In particular, the client strongly prefers to work offline, so our previous system design that bypassed Zotero and edited the cloud storage via the PyZotero would no longer satisfy the client's needs since it would require an Internet connection. Thus, the developers and business teams met and have agreed to restructure the project in order to accommodate offline use.

Changes & Notes:

1. We have switched from an application to a Firefox plugin. The reason is simply because Zotero does not recommend directly editing the local SQL Database directly as it can result in corrupted data when syncing with the online database, Zotero instead recommends use of the JavaScript API located in the standalone/Firefox/Chrome directories. Unfortunately the API was designed with the assumption that whoever is using the API is developing a browser based plugin, hence our switch to a FireFox Plugin. Our plugin works with FireFox and the Standalone.

2. Our old python code which uses the PyZotero API still works fantastic for users that will be using Zotero while connected to the internet, therefore we will make it available as a set of tools for other developers or interested users.

System Design Explanation:

- Our Plugin GUI now uses XUL files instead of Python 2.7.10, just as before the user will interact with this part of our plugin to perform tasks.
- Our Plugin Backend has changed from Python 2.7.10 to JavaScript to better interact with the JavaScript API provided by Zotero. It takes and processes input from the GUI and through firefox interacts with the JavaScript API.
- FireFox, needed to expose Zotero Internals (i.e. the JavaScript API) to our backend for use.
- Our Plugin now uses the JavaScript API instead of the PyZotero API, this has the advantage of making changes locally. The Zotero Online DB will be updated with any changes made offline, or allow the user resolve any conflicts between the offline DB and the Online DB in a similar way to git or other version control systems (this is a built in Zotero function).
- The Local Zotero database (DB) is a SQLite database that Zotero Standalone/Firefox/Chrome uses to store all user data (through the JS API).