DELIVERABLE #3

University of Toronto Scarborough

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

Date : Oct 26, 2015

Table of Contents

1 Current State of Project

The first week of our sprint #1 was used primarily to test the waters with the Zotero source code and how to create a plug in. We were plagued with assignments from other courses and unfortunately did not complete as much of our sprint as we wanted in the first week. We focused on creating the build during reading week but had a difficult time building the zotero-standalone, which slowed down the development. We then spent a long time trying to find a way to complete our user stories through the client source code, but the sparse documentation and dense framework made it difficult to make much progress. We finally decided to settle to make a plug-in using Pyzotero (an API wrapper in python for the Zotero API) after meeting with our teaching assistant and realizing that it was inadvisable to modify the source code directly.

Our meeting with our teaching assistant also showed us that one of our user stories was more difficult then originally anticipated and too costly to create compared to other user stories of similar priority. We quickly gathered for an emergency meeting to decide on how we want to approach this ordeal. At the meeting, the developers of Precision Development explained the problem to the business. It was deemed that to meet the requirements of the feature to create custom fields, we would have spent too much resources implementing dependent components of the feature. For example, since we cannot alter the Zotero cloud database, we would have to create our own database that would have the same sanitation features, the same security features, etc. In total, the cost of implementing this feature with our current design was grossly disproportional to the cost of other user stories with similar priorities. After explaining in detail the risks and time constraints, the business decided to change the user stories for the first sprint. Thus, until we can create a new design, we have switched the user story for creating custom fields for a record (user story id #3) to coupling duplicate records in groups (user story id #2).

Following that, we spent our last 6 days grinding out the developer hours we missed from the first week. We decided to create a plug-in instead which had a UI separate from the standalone client. The plug in currently has the batch editor feature with most functionality tested, and a partially created UI for creating custom bibliography styles (user stories #1 and #4 respectively). Coupling documents feature (user story #2) has the backend methods implemented but is not linked to the UI yet.

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	40 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	Low	50 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	50 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	40 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 #1

3.1 Sprint Backlog

Our sprint backlog is located here:

https://docs.google.com/spreadsheets/d/1x4ZO0lsaXktc8lxdIhvAAT5QEFSecOz8eK0PieLRWok/edit?pli=1#gid=0

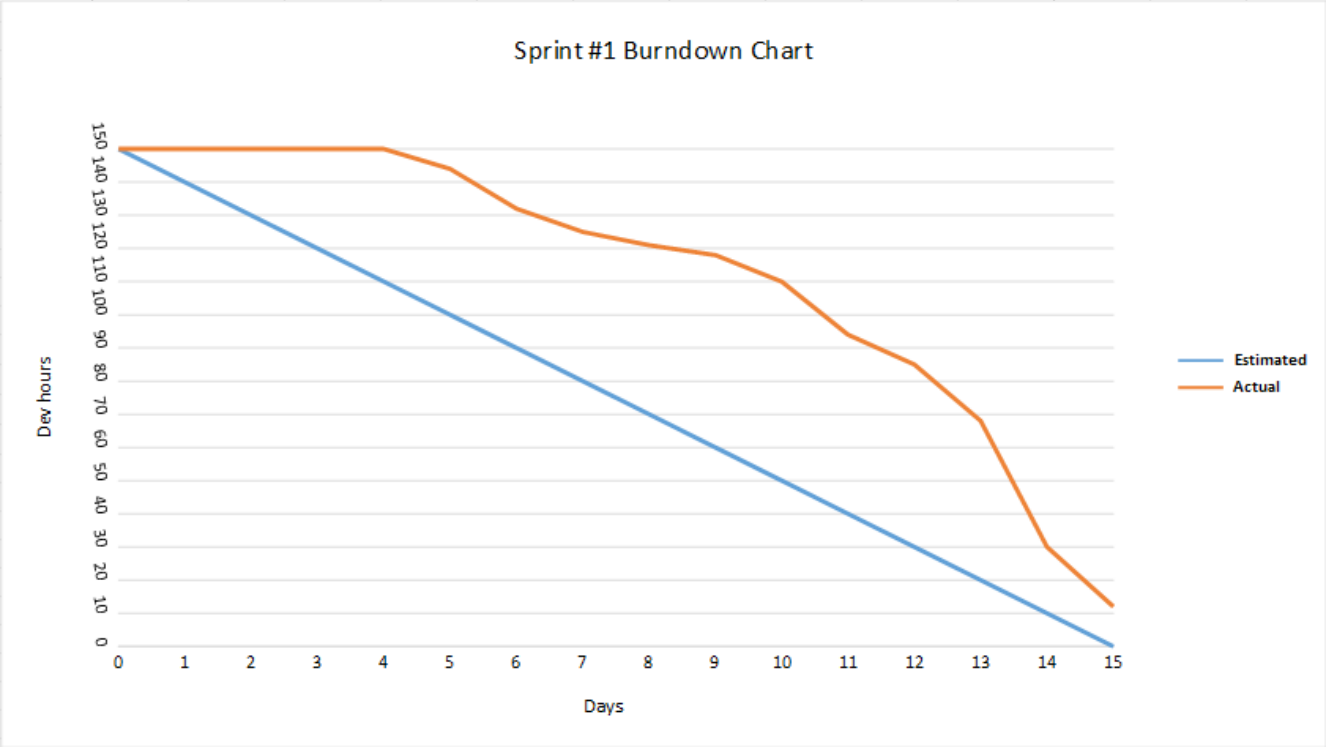
3.2 Sprint Task Board/Plan

Our task board has been moved to Trello because it provides easy to use management tools that organizes the tasks nicely.

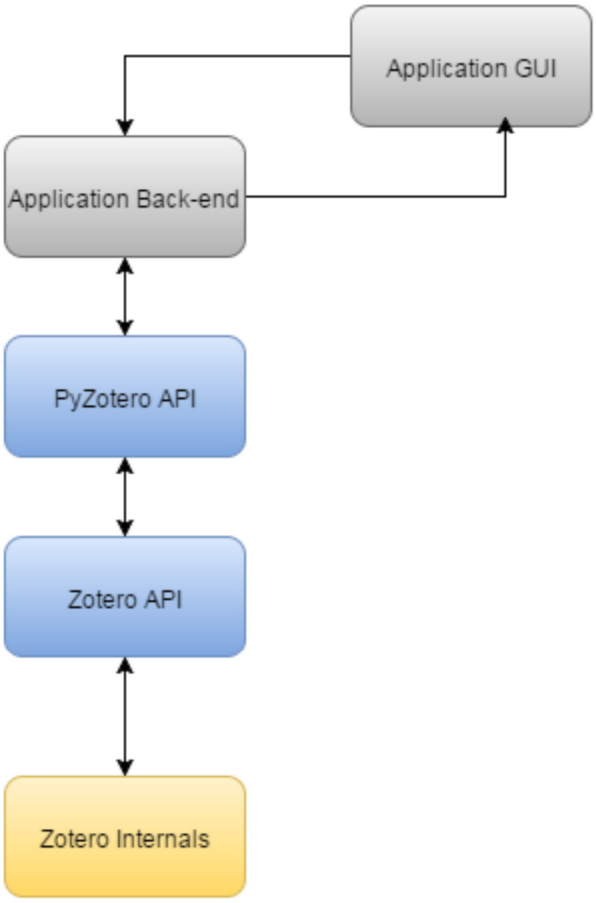
Our task board is located here (snapshots are located under "INSTRUCTIONS"):

https://trello.com/b/bA6DMjMR/sprint-1-task-board-precision-development

3.3 Sprint Burndown Chart



4 System Design Diagram



Explanation of System Design:

We are making a Python (2.7.10) application that will, through the API's, Interact with Zotero in order to make changes to the user's Zotero Library.

- The Application GUI is a Python GUI that the user will interact with in order to use our application. (Features can be found in our sprint backlog/taskboard/reports) The GUI then sends this information to the App. Backend for processing.

- The Application Backend processes information from the GUI and calls the PyZotero API to receive, modify, and update the user's Zotero Library. This process requires an API key that the user will have to generate and provide in order to use our app. The backend will then update the GUI accordingly.

- The PyZotero API (written in Python 2.7.10) is a python wrapper for the Zotero API, that our backend interacts in order to gain access to and modify the user's Zotero Library.

- Since our design doesn't ever interact directly with the user's library our design puts things such as the user's library, Zotero Auth. Server, Zotero DB, etc. under "Zotero Internals"

tl;dr

Our Python App. uses the PyZotero API to perform various functions on the user's Zotero Library.