

COMP4920 Project Plan - Bookswapp

Group 4: Adrian Groch, Alex Nguyen, Michael Su, Michelle Lu

Project Description

Bookswapp is a web app for university students to buy, sell and trade textbooks. Users will be able to search all book listings, add books they would like to sell for a price, or which they are willing to trade for another book. Users who register for an account will also have a personal wish-list, to which they can add books they are looking to buy or trade. When books matching this criteria are added, a notification will be sent.

Product Backlog (Prioritised, estimated and including MVP)

Each item has been prioritised on a scale from 1 to 5, indicating lowest to highest priority. They have also been estimated using story points on a fibonacci scale (1, 2, 3, 5, 8) to indicate their relative complexity. Task hour estimates, based mostly on the complexity estimate, will be done for each sprint during planning for that specific sprint. This allows the task hour estimate to be adjusted based on estimates done during previous sprints.

The minimum viable product includes all user stories or tasks with priority 5, and it has been highlighted in the product backlog table on the next page.

Frameworks and Tools

The front-end will be a single page application implemented using Bootstrap for CSS and AngularJS for responsiveness. Testing will be done using Protractor for end to end testing, as well as user testing. Wireframe mockups will be done using Balsamiq (see Appendix).

The back-end will be implemented in Python using the Flask framework, and MySQL will be used for the database.

Github will be used for version control and Zenhub, integrated into Github, will be used for agile project management.

Any library dependencies used will be open source with unrestrictive licensing to minimise risk of infringement or litigation on release. Flask (BSD-3), Flask-login for session management (MIT), AngularJS (MIT) are examples of this.

All team communication will be conducted on Discord, a voice and text chat program, and face to face meetings.

Table 1. Final Product Backlog and Planning poker results

Epic	Feature	User Story	Priority	Michelle	Michael	Alex	Adrian	Estimate
Allow a user to make a textbook transaction based on the inventory available on BookSwapp (post a book for sale or swap)	Main page containing all books on app	As a user, I want a homepage so that I can see all available books	5	8	3	5	8	3
	Contact details exchange when buy offer accepted	As an authorised user, I want to exchange contact details so that I can meet up with the buyer	5	3	2	3	2	2
	Attach an image to a book listing	As an authorised user, I want to add an image to a book I am listing	4	1	1	1	3	1
	Setting a price or margin you are willing to sell a book for, or an equivalent book that you are looking for	As an authorised user, I want to set a price range for a book I want or a book I am selling	4	3	5	3	3	3
Allow a user to navigate the books available on BookSwapp efficiently (by search, user, all available)	Being able to add new books for sale or swap	As an authorised user, I want to add a book so that I can sell or trade it	5	5	5	5	5	5
	Removing books for sale/swap	As an authorised user, I want to remove a book I've listed	5	1	1	1	2	1
	(Search function) Able to lookup database of books based on ISBN, name or course, university, author	As a user, I want to search for available books	4	5	5	5	3	5
Allow a user to manage their	Being able to register a new	As an unauthorised user, I	5	3	3	5	4	3

account	account	want to register an account						
	Logging in/out of account	As an authorised user, I want to logout of my account	5	3	2	5	3	3
	Being able to delete an account	As an authorised user, I want to permanently delete my account	5	1	2	2	1	2
	Being able to edit your profile details	As an authorised user, I want to change my account or profile details						
	User book Wishlist	As an authorised user, I want a wishlist so that I can add books I currently need	4	3	5	3	2	3
Provide an intuitive user interface & experience to guide transactions (single page, fast hosted web app)	Research databases		5	2	2	2	2	2
	Research front-end framework		5	2	2	2	2	2
	Research testing tools		5	3	3	3	3	3
	Storyboard app frontend - general layout/style to be applied to every page		4	5	8	5	5	8
	Being notified when a new book enters the database that matches your criteria or needed books	As an authorised user, I want a notification when a book on my wishlist is listed	3	5	5	5	3	5
	Deployment as web app - hosted on cse servers		3	2	2	2	2	2

Risk Management

Table 2. Risk Management Summary

Risk	Mitigation
Overcommitment (in project timeframe)	<ul style="list-style-type: none">• Review and re-evaluate progress at end of each sprint• Once planned, sprint backlogs will be fixed/unchangeable to prevent scope creep
Infrastructure	<ul style="list-style-type: none">• Mockups/Wireframing• Restful API• Deployment via home server
Technical roadblock	<ul style="list-style-type: none">• Devote sufficient time to research• Open to further research if required• Sprint plans do not encompass mid-semester break; giving time to reach out for help overcoming obstacles
Financial difficulties	<ul style="list-style-type: none">• Use of open source libraries with unrestricted licences (Angular under MIT licence, Flask under BSD-3 licence)

Release Plan

Release v0.0 - Proof of concept (September 15 - September 21)

This first release will be based on implementing the bare framework for the application, that we have a database set in place which is working, our skeleton API is interacting with the database and the frontend website is interacting with the API.

Release v0.1 - MVP User Stories (September 22 - October 5, including mid-sem break)

This release will focus on implementing all the main functionality that will support all the cases of our user stories, allowing us to achieve a minimum viable product.

Release v0.2 - Priority rating 4 (October 6 - October 12)

This release will contain all of the features that we deemed as a priority level of 4 - more than just the minimum viable product, we wish to have features that make our web application worthwhile, resulting in a pleasant user experience.

Release v1.0 - Anything else feasible (October 13 - October 19)

This final release will contain all of the other features that we will have time to implement. Since it is not feasible to assume that everything will go to plan over the course of the project, we have given ourselves this buffer in case we were not able to reach certain goals by due dates.

Project Due (October 23)

Buffer time for tasks to go over initial estimate.

Sprint Plan (for initial sprint)

We analysed the complexity compared to priority of each and agreed on a time commitment estimate from everyone. The results are below, with testing given extra time given that it will be a key in ensuring final product quality. First week will involve some feature stories focused on learning the frameworks and enabling session management and a basic populated book page UI.

Table 3. Sprint 1 plan

Task Description	User Story	Priority	Complexity Estimate	Task Hour Estimate
Main page containing all books on app	As a user, I want a homepage so that I can see all available books	5	3	2
Being able to add new books for sale or swap	As an authorised user, I want to add a book so that I can sell or trade it	5	5	3
Being able to register a new account	As an unauthorised user, I want to register an account	5	3	3
Logging in/out of account	As an authorised user, I want to logout of my account	5	3	3
Removing books for sale/swap	As an authorised user, I want to remove a book I've listed	5	1	1
Research and implement databases		5	2	2
Research front-end framework and set up base main page		5	2	2
Research testing tools		5	3	4

Scrum Roles

All members of the team will be developers, and the scrum master and product owner roles will be additionally assigned and rotated for each sprint. At the conclusion of the project, each team member will have had experience in all roles.

Table 4. Plan for scrum role allocation

Week	Scrum Master	Product Owner	Developers
8	Alex	Michelle	Adrian, Michael
9	Michelle	Adrian	Michael, Alex
10	Adrian	Michael	Alex, Michelle
11	Michael	Alex	Michelle, Adrian

Scrum Master role:

As a guideline, the scrum master for current sprint will:

- Initiate daily standup and group meetings
- Responsible for communicating the outcome of planning meetings to absentees
- Tracks status of ZenHub board to see how developers are travelling with backlog
- Check workload is reasonable for each developer when assigned and work with the product owner to make sure developers don't overcommit
- Works with product owner at some point each the sprint to check & make sure there is **prioritised** stuff in backlog for next sprint to do.
- Checks on sprint burndown/progress when necessary, makes recommendations on improvements for next sprint

Product owner:

The product owner is:

- Responsible for the user's perspective - steers the final project backlog
- Controls feature backlog priority (make suggestions for what should be done next and check with Scrum Master if feasible)
- Checks with client (John, our seminar tutor) for feedback on feasibility and perspective on feature importance (once MVP is released)

Meeting Schedule (Daily standup, sprint planning, sprint review, sprint retrospective)

Daily Standup

Our daily standup will be held on discord since it seems illogical for all of us to meet in one location as we all live around various parts of Sydney. As with any good stand up, we're attempting to keep it less than 5 minutes long, preferably around 9pm as we have all finished uni by this time and are usually active online. Furthermore to keep the standup short and to the point, we will all answer the following three questions to the group:

- What did I do yesterday?
- What will I do today?
- Is there anything in my way?

Sprint Review

Sprint reviews will be held in-class on October 5 and October 19.

Sprint Planning and Sprint Retrospective

Our weekly sprint retrospective will take place on the end of our tutorial on wednesday at 6pm. We chose this time because it's a given that all group members will be there to attend unless specified earlier.

We will reflect on the sprint that just ended by asking each other to reflect on the past week, highlighting troubles and issues that we ran into along with aspects and areas that we think we did well in. We will use this information to improve the next sprint in the coming week.

Relevant Extreme Programming Practices

Test Driven Development (Back-end)

Unit tests will be written before programming. For back-end code linking front-end with database, Python unit testing modules (unittest) will be used to test connection integrity and content retrieval where possible.

User Feedback and Integration (Front-end)

Since the basis of our web application is to provide a user friendly experience for people wanting to buy, trade and sell textbooks, it's important that we develop our application based on the feedback we receive from our users. This will ensure we're building something that fulfils the criteria that the target audience is wanting and expecting.

Pair Programming

In the early stages of development, we will be using a pair programming approach. As the team will be broken up into both frontend and backend, it's important that everyone understands how the application interacts as a whole. Once everyone has an understanding of the code base, we can then work on making the application modular, allowing development to be undertaken individually.

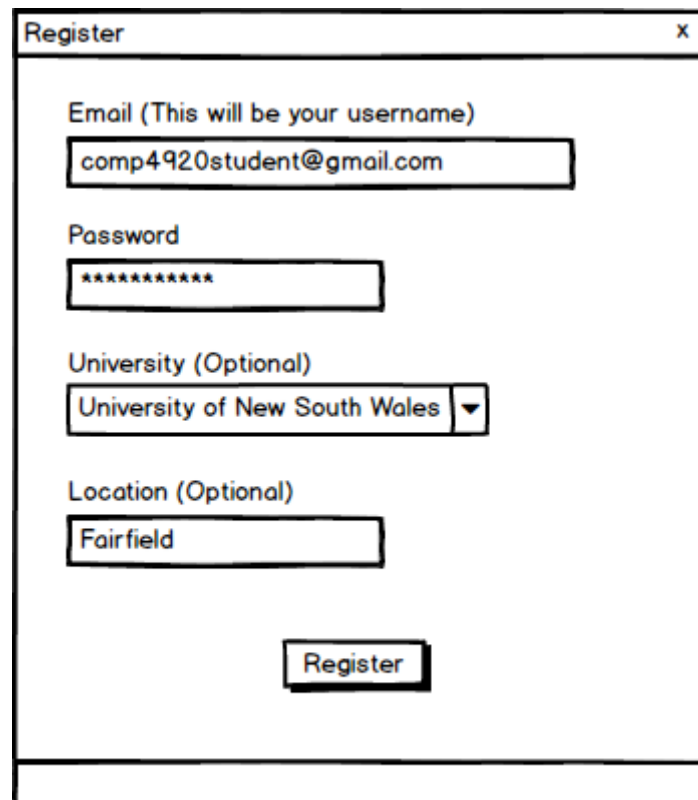
Planning poker

We will make use of planning poker during weekly sprint retrospective meetings to update our predictions of what can be achieved as the project unfolds. We found this effective from applying planning poker for the initial sprint (see Table 1 and 3 above) as it ensures everyone agrees on expected output by sprint end.

Continuous integration (using git, managing pull/push requests on updates)

Using Git as a version control system will ensure all developers will have the latest source code available and will be notified of conflicts in code as soon as possible. We will make sure to commit frequently to minimise the chance of redundant code written by developers working in parallel. Tests will be run with major commits in terms of code logic where possible and general front-end issue scoping from the browser.

Appendix



The image shows a wireframe of a 'Register' window. The window has a title bar with the text 'Register' and a close button 'x'. Inside the window, there are four input fields: 'Email (This will be your username)' with the value 'comp4920student@gmail.com', 'Password' with masked characters '*****', 'University (Optional)' with a dropdown menu showing 'University of New South Wales', and 'Location (Optional)' with the value 'Fairfield'. A 'Register' button is located at the bottom center of the form area.

Created with Balsamiq - www.balsamiq.com

Appendix 1. Mockup of the account registration window

http://www.cse.cuhk.edu.hk/comp4920/comp4920.html

Login

Username (Email)

comp4920student@gmail.com

Password

Login

[illegible]