COMP3900-H15A-capSquad - Retrospective B Report

Daniel Latimer
z5115175

Connor O'Shea
z5115177

Kevin Chan
z5113136

Oliver Richards
z5157383

Peter Kerr
z5115807

November 6, 2020

Contents

1 Introduction

Our team's progressive demo A was held on the 5th of November at 1505hrs. At 1530hrs on the same day, our team held a retrospective meeting. All team members were present at the retrospective. The following document outlines the overall reflection of the team, including a reflection on our actionables from the last retrospective, as well as actionables for the final demonstration.

2 Things we did well

The main thing that the team was happy about was that most of the functionality of the application had been completed. At this point, a few user stories relating to the UI for novel features remain, and they can be completed within one sprint. Furthermore, the group members who have been working on the backend introduced unit testing, which has kept the API relatively stable. An additional benefit of unit testing the backend was that the frontend team ran into less API errors while developing.

For teamwork, we found that Git management had gone quite well, as there have been minimal merge conflicts. We believed that this was because the work was distributed between team members well, so nobody was working on the same part of code at the same time. Finally, we have found that our work has followed the Agile methodology quite closely - a working prototype has always been available and it has gradually incorporated more user stories.

3 Things we could improve on

Most of the discussion in this section was related to the Demonstration B presentation. The team member who was presenting the application had slow internet, resulting in a lag between what was being said and what was being shown, making it hard to follow. To add to the problems, playing podcasts over BlackBoard Collaborate was quite soft; which we attributed to the slow internet as well. Furthermore, we spent a lot of time switching between two accounts to demonstrate the uploading podcast feature - time we could have spent on going through Jira user stories in more depth. Finally, when attempting to unfollow a friend account, a large error popped up on the screen. This happened because the frontend server was running in development mode rather than production mode. If the server was run into production mode, the error would have still occurred, but it would have happened without disrupting the presentation.

Beyond the presentation, the database lost integrity (records had mismatched fields or were missing) a couple of times in previous sprints. This was attributed to constant changes in the database schema. Overall, a plan has been made to address problems with the presentation and the database, which are outlined in the following section.

4 Things to try

The table below outlines a list of tasks the team has agreed to try in the next sprint.

Task	Reason	Member
Practice final demonstration	We need to practice the final demonstration to become familiar with the presentation setup.	Everyone
Using production and build servers	Production servers suppress ugly errors from showing up on the screen.	Kevin
Create a 'production' database separate to the backend development database	Having a different database to the development one prevents accidentally pushing breaking changes to the frontend branch and to prevent a database from losing integrity.	Oliver

5 Retrospective A Effectiveness

In the sprint after the retrospective A meeting, we endeavoured to fulfill the responsibilities actioned from the meeting. A summary of the results are below.

- *Using Django as our backend server*: investigated moving to a Django backend but it appeared to introduce a lot of complexities. Solving the problems we had with user authentication and filtering GraphQL results within the Flask framework ended up being easier than transitioning platforms.
- *Implementing a business logic layer*: good for adding all the extra logic surrounding users - (un)following, (un)subscribing, etc., so ultimately it worked out well.
- *Assigning a manager for the recommendation system*: adding a recommendation text channel greatly streamlined communication regarding the functioning of the recommendation system. Having one person overseeing the development of the functionality meant that it was easy to pipeline all communication through one person, meaning that there was no wasted
- *Streamline design*: agreed to storyboard new pages using Figma, which focused the UI of the website.
- *Updating user stories*: user stories were modified to fit within one sprint. This helped greatly with predicting how much work could be finished within a sprint.