COMP3900-H15A-capSquad - Retrospective A Report

Daniel Latimer
z5115175

Connor O'Shea
z5115177

Kevin Chan
z5113136

Oliver Richards
z5157383

Peter Kerr
z5115807

October 15, 2020

Contents

1 Introduction

Our team's progressive demo A was held on the 15th of October at 1500hrs. At 1600hrs 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 as well as a list of things to try in the next sprint.

2 Things we did well

Firstly, the team thought that most of the system architecture was good. Using MongoDB, GraphQL and React allowed us to gain experience in modern, popular technology. An additional benefit was that these components were well documented, so we could utilise their tools to help integrate the frontend, backend and database. Furthermore, our team thought the UI was clear - the podcast player at the bottom of the screen was a notable feature.

In terms of teamwork, everyone agreed that the level of communication was good and that we had been achieving the right level of team conflict during meetings. Additionally, we thought that week-long sprints worked well for project planning and personal schedules. Finally, we found that our main communication channel (Discord) and the sprint management system (Jira) improved project visibility among team members.

3 Things we could improve on

Unfortunately, it seemed like we made a poor choice in our web-server technology, Flask. Flask lacked proper integration with GraphQL, making it hard to understand how to utilize it, especially with developing GraphQL for the first time. On the frontend, the UI lacked explicit labels and titles, and did not have a centralised design pattern. For example, the audio controller at the bottom of the screen was not integrated with the "recently played" page, resulting in the user being able to play two podcasts at once. On the backend, a clear business logic layer was not made, resulting in some code duplication. Finally, the team agreed that the user stories made were too big and encompassed more work that could be done in one sprint.

4 Things to try

The table below outlines a list of tasks the team has agreed to try in the next sprint. Each team member is responsible for enforcing one of these tasks. An outline of the effectiveness of these tasks will be presented in the Retrospective B Report.

Task	Reason	Member
Researching Django as an alternative to Flask	Flask introduced some pain points in previous sprints, so we want to see if Django is a reasonable alternative to move to.	Oliver
Introduce a business logic layer	A business logic layer on the backend will allow a clear place to develop business rules and prevent code duplication.	Connor
Oversee the recommendation system tasks	The recommendation system is the most complex feature of our application. To develop this feature as fast as possible, we want one person to oversee all tasks and enable communication between team members to complete subtasks.	Peter
Streamline design of UI	UI lacks a centralised design pattern. By streamlining all UI, the website will look and feel better.	Kevin
Revise user stories	User stories need to be smaller so that they can be achieved in one sprint.	Daniel