A screen shot of a computer screen

Description automatically generated

Software Engineering 3

project retrospective/review document

Eoin Geoghegan | Dimitrije Sreckovic | Ciarán Tone | David Ward

# Introduction

This document serves to outline and review the processes and development considerations that were undertaken in the development of the newsagent application made for the continuous assessment portion of Software Engineering 3.

## Roles

Everyone in the team was responsible for certain aspects of development of the application. There was no hierarchical setup, rather, each individual was to design, test and develop the various components that would be needed to achieve the brief. Practically this meant that each worked on the following entities/area/functionality:

* David – Delivery Area
* Dimitrije – Database functionality
* Ciaran – Publications, Command-Line Interface
* Eoin – Command-Line Interface, Customer, Order

## technolgies and project setup

Once the brief was understood to a reasonable degree, we set out to decide on the software development lifecycle to use. Initially there was debate over whether to adopt a Waterfall approach to the project as it was felt that this was an easier way to present a strong plan that we could stick to week-to-week. The ability to see the project as a whole and work from a pre-set blueprint was particularly attractive.

The counter to Waterfall was that an agile approach in the form of Scrum would enable us to be more suited to change. We anticipated that, given the lack of technical understanding from this hypothetical customer, new requirements could emerge during development. Additionally, agile lifecycles appeared to offer a better ability to manage situations where we may not have accurately estimated the development of a feature. By going down the agile route, we were allowing ourselves the opportunity to not have unforeseen developments have a significant or compounding effect regarding delays to the project.

Regarding software, GitHub was the version control system of choice. Its reliability, flexibility and ease of use enabled us to begin considering design, project planning and other crucial matters without being delayed with setting up tools. Beyond tracking commits to the repository, progress was also tracked via the use of Taiga – an open-source project management tool – wherein user stories and their associated tasks were displayed for all to see and manage. As with GitHub, changes here also notify everyone on the team.

# Challenges & Issues

In the first sprint there were issues regarding integration at the end. In line with typical agile practices, we endeavoured to produce working software at the end of each sprint. While we improved in the second and final sprints, the first sprint was messy in that, on the final day of the sprint, conflicts and other merging issues hampered our ability to display a working demo, though this was rectified the following day.

The integration issue was a visible highlighting of time management and estimation issues that we faced continuously throughout the project. While drastic improvements were observed over the semester, the team would overcommit or - in response to missing the mark - under commit in terms of what could realistically be achieved week-to-week. Given the busy nature of a college semester, it was imperative that time was adequately allocated and effectively utilised in each week to ensure smooth progress was made. Overall, we believe we worked well in this regard, however there was scope to fine tune our workflow and produce more than what was finally delivered.

Better estimation and time management could have been achieved through better communication. The systems we utilised (Taiga, Discord, GitHub notifications, etc.) to bring one another onto the same page, while excellent in themselves, arguably were not used to the fullest. There were instances where we would reach the end of a sprint and there were still some areas where confusion would arise into what was expected from everyone.

A strength of agile teams is the review that occurs after sprints are completed where teams can evaluate their efforts and see where there is room for improvement. The team would convene on a weekly basis to discuss the sprint, like the daily stand-up seen in industry settings. A challenge we encountered was using these meetings, not explicitly for review, but for advancing the project from a purely technical perspective. While we do not believe that gathering to discuss design, prioritisation or development, we acknowledge that more time could have been allocated to examine ourselves retrospectively to find weak areas that could be improved on collectively.

# Retrospective Insights

Despite the challenges we faced and pitfalls we encountered, we believe that we nonetheless worked in such a way that we can be proud of. From the outset we established the tools that would enable us to work at our best, and defined a working agreement that would clearly state what we could hope to expect from one another. Establishing this strong foundation of resources and responsibility enabled us to do better than would have otherwise been possible.

While already familiar with the theory surrounding Scrum and Agile project management, we were thrilled to utilise such a lifecycle in practice. We are content that we learned a great deal and recognise where there is room to grow, though we believe we excelled overall. We were able to gather requirements early on and craft our product backlog (and by extension sprint backlogs) very quickly and did not find ourselves re-visiting our past steps. Everyone was able to see what their responsibilities were and never failed to deliver for the team especially where software dependencies arose. Code was well-tested and sent to the version control system for everyone to make use of without issue. While inexperienced in test-driven development, everyone adopted it well and produced great work as a result.

# Final Remarks