Milestone 4

Testing Implementation Update

We have implemented unit tests via NUnit (a .NET testing framework) and BUnit (a Blazor testing framework) that allows us to simultaneously test control logic, the functionality of the UI, and the control logic related to the UI. We have agreed to a collective goal of attaining an 80% statement coverage rate for all work. As part of testing, we have implemented the use of mock classes and objects. An unfortunate circumstance is that we are not always able to meet this coverage goal as there are certain functionalities we cannot feasibly mock.

Overall, though we are thorough with our testing, we have found that implementing testing has come with a lot of testing bugs occurring, forcing us to deal with those before we can continue implementing other use cases. Furthermore, testing hasn’t been as effective as we hoped at revealing issues.

We did try to set up continuous integration on Azure, however our free tokens expired and so we’re unable to use it without having to pay out of pocket. Thus we have opted not to launch it.

Functionality Testing Update

At the moment all the functionality that has been fully implemented has been tested in accordance with our testing plan. At the moment we have completed and tested the following:

1. Registration of new users and Login of existing ones including authentication and encryption of passwords
2. The Home page and Navigation Bar
3. The ability for existing users to update their profile details including their username, email and password and avatar
4. The sending of chats between different users
5. The ability of users to add, remove and assign permissions in groups

Automation

At the moment, we have started using playwright to help automate our testing especially of front end functionality but we have had some trouble setting it on different machines. However, it does allow us to turn user actions into tests very efficiently.

Project Summary

At this point we’re almost done with implementing all the base functionality of the project that we hoped for. As for a process, we never really selected one and instead focused on trying to fulfill the requirements of scrum and kanban. We have found that the scrums were segmented in such short, fast paced sprints that did not consider the full course load that most students have outside of this project. Furthermore, most of the time we spend far longer than 8h to complete our issues for the week to make sure our issues would get completed. We found this was also partly due to us only starting to create the project in Week 8. As such if we were to redo this project we would likely try to complete documentation early in the semester and start coding, as well as work with 2 week scrum cycles.

As for ensuring the quality of the source code, we believe that having multiple reviews on each pull request has allowed us to identify any code smells and ensure they aren’t pushed to the main branch. Furthermore we have implemented several design patterns including factories, events, observers, publisher-subscribers and codecs. These design patterns have helped us ensure there is minimal coupling and maximum cohesion between components. Furthermore we have delegated most of our data management tasks to services rather than components ensuring our code is modular and scalable.