Team PorkBelly
Sprint 2 Retrospective
8th October, 2021

Retrospective met at: 4:14pm

What was done	What went well	What did not go so well.
Tasks outlined in Sprint plan.	We stayed on track. Did not have any significant delays. It was impressive how the visual styling task was split up and handled by the team members.	Working in CSS was found to be frustrating, team members would like to have more familiarity through reading CSS standards, MDN references, and W3Schools examples.
Modal for confirmation for deletion, errors and warnings.	Pair programming on Discord calls provided a casual, 'same room', environment for those present.	Current front-end testing is not useful. The Enzyme library could be used to logically render functional components for unit tests. Taiko still remained as the primary end-2-end testing suite.
Prototype Integration Testing CI		Visual style's size was underestimated as at that time since at that time the work was thought to be done by two people.

Discussions:

Walter began the retrospective session with listing out the completed tasks (through GitHub issues). He then proceeded to ask how the other team members felt of their sprint.

Richard found that the styling task's size had been heavily underestimated. He found himself frustrated while doing the task. 'Can't be helped. It was just work.' He found the image parallel processing task a challenging but fulfilling feature. Parit concurred with Richard's struggles. He learnt a lot from styling in CSS on FluentUIs component. Through learning from the code of other members did he then develop a feel. The team styled the FluentUI using an in-house theme to define styles for its component. Walter asked Richard whether there could be something that could be done to reduce these frustrations. Richard replied that it was more the nature of CSS and its implicit rules, but Shang Zhe added that he would get more familiarity through reading CSS standards. He stated that HTML and CSS provide useful features well suited for normal document flow, e.g margin collapsing feature.

Walter said that he had noticed Richard and Yujian regularly engaged in pair programming, and asked how they found it. They replied that pair programming provided a casual environment where they felt like they were at the same table, able to occasionally toss jokes.

Walter asked how Parit had found the sprint. He had struggled with CSS, and felt that the time he spent on using it could have been spent elsewhere. Walter said that he had found MDN to be a great resource for learning CSS, to which Shang Zhe concurred that it was probably the best human-friendly source for getting as close to the Standard as possible. Parit added that he found W3Schools to be useful for providing examples to study. Yujian agreed with W3Schools. However, Shang Zhe stated that MDN provided an in depth explanation to tackle fringe cases as well as refers to the standard. Richard added that there were some cases for CSS that sometimes MDN may not provide.

Lastly, Walter asked Shang Zhe of his opinion. He found that the teams workflow had been in sync especially with how the styling tasks had been split into modular tasks; it felt like an indie development team. He noted one drawback, that the front-end tests do not provide a means to sniff out bugs. As the testing script for react does not test interactions, the team was left with using Taiko to automate browser testing. Richard asked whether a testing suite for react components exists. Shang Zhe referred him to the Enzyme library which could render component in a logical fashion. This could be used to used do unit tests on the components -- allegedly FluentUI employed this testing suite as well. It would be better than the snapshot tests right now.

Richard asked Shang Zhe about using JEST to test the webworker logic. He replied that the implementation of the webworker should be extracted into a function and independently tested on JEST.

Richard also commented that the decomposition of the styling task had not received as size estimation. He mentioned that during the time of size estimation, they had planed for two members to work on the whole, rather than to split the job.

Plan for next sprint:
In alignment with the Client's decision, the team would work on testing and then implementing favourite cards feature. Richard was confident that implementing favourite cards was a simple feature to carry out and promised to have it done quickly. He would add a checkbox to the Card Details Panel, enforce that favourite was send through to the back-end, and finally establish the logic to prioritise favourite cards in the Card List component.

Shang Zhe recommended tackling integration testing required for the Acceptance Tests.

Richard suggested using Taiko to complete the acceptance test through an automated browser tester. Shang Zhe took the task of writing a proof of concept Taiko test for AC1.

The feasibility of using Taiko for testing was discussed. Walter stated that he would be fine with carrying out Acceptance Tests manually, but Yujian stated it could overlap with the purpose of usability testing. New bugs and usability issues could be gathered from those tests. He hinted that these issues could be the starting point for extensions of the project, and that the team should not stress out on meeting them. The team could discuss usability tests another time but so far the interviewing process carried out by Yujian at Sprint 1 was good.

Richard and Walter then engaged in a long discussion on the strategy of implementing the Taiko tests, especially due to the repeated preconditions of many of these Acceptance Tests. They learned from Shang Zhe that it was not possible to 'Clone' a Taiko instance to create a copy of the state of the 'Browser' in a testing session. They also learned that each individual test would be carried out in parallel as issued by JEST. He recommended that these tests should be organised into a directed acyclic graph (DAG -- e.g a tree), then topologically sort to identify which tests to create first.

Richard and Walter wished for a strategy that reduced code reduplication in regards to achieving the necessary preconditions and data samples used for testing. They also found that each precondition itself is a result of a test, and then formulated the current (to Walter's understanding) plan: the tests could be sorted into a tree, whose leaves are the tests that would decide on a pass or fail. Each non-leaf node in the tree outlines a 'Procedure' that navigates the Taiko instance (each instance will have to begin from a logged out state), to the desired precondition, akin to traversing down the tree, for the test.

Each testing procedure would then be a path from the root of the tree down to one of its undiscovered leaves. Each procedure may be defined as an exported function taking arguments for its Taiko procedures, and ensures the resulting state of the Taiko instance.

Size estimation for the upcoming sprint (focused on testing and the favourite cards feature) was postponed until Shang Zhe's Taiko pilot test was completed.