Team PorkBelly

Meeting minutes 24th September, 2021

Meeting commenced: 2pm
Attendees: Walter, Parit, Yujian, Richard

Talked about parent callout not dimissing inner dialogue on closed. Tried to change onDimiss of TagEditor to onBlur – Blurring any child element of the dialogue triggers the callout onBlur.

Other bugs were showcased:

1. Deleting a Tag currently attached to a card throws an Error and crashes the site.
2. Calling app.newTag() or Tag.delete() refreshes the application state and reverts all unstaged changes to a card.

These bugs would be published on GitHub to broadcast to the team.

Yujian brought up an issue with the extending TagButton to update the TagQuery. On inspection of the implemented AppContext in App.tsx, the update function, at the time of discovery, did not include fields for updating the tagQuery, nor was there a search query state in the App.

Richard stepped in to implement a quick solution to this problem. There was a short discussion on the discrepancy in TagQuery in IAppContextProperties and IAppContext.

ShangZhe joined 2:55pm.

ShangZhe advised that for each re-render of the application, there will be fresh implementations to both Tag and Card controllers because of the renewed user state. Hence, updating the tagQuery may be a little more nuanced. He offered to extend the update function.

This was done by extending context.update to accept a list of tag ids, via the tagQuery field and setting the tagQuery context to map these ids to implemented ITag's on the context.

Walter described the new bugs he discovered to Shang Zhe. Then they discussed the current problem with nested tags causing an improper event capturing. Shang Zhe suggested having the TagWrappers set the state for the currently focused tag's HTMLId, and to extract the callout from the hierarchy.

Richard discussed his 'auto-scrolling issue' where he wanted to recorded whether a card was required to be locked in the same y-coordinate on the screen when the card detail panel is open and or expanded. His concern was when the user scrolled along the card list, whereby this lock to should be release. The goal of this feature was to maintain a scroll-percentage despite screen size and detail expansion.

ShangZhe asked whether the client requested for a mobile view, as he would be implementing the visual styles. Yujian and Richard both recalled that the client wished for a Web Application first.

Discussion returned to maintaining the scroll percentage. Shang Zhe wanted to request why knowing the viewport size is required, Richard intended to use this value to determine whether Home needs to re-render to maintain a fixed scroll percentage. He intended to have this percentage set during onScroll events and have Home re-rendered with the resizing of the component.

Richard wants to be able to programmically scroll the card list without changing the scroll percentage, and he would need to iron out these uncertainties.
ShangZhe notified Walter that he would have the bugs fixed, before diving into visual styles. He asked Yujian for his progress (issues #86 #85). He reported that he needs to hook the update tag query to the TagList Component, as well as to consume the TagQuery on the Home.tsx

Parit would start on issue #87 to complete it, this can be done concurrently to styling.

Regarding testing, ShangZhe suggested having a mongodb server running on the continuous integration, or alternatively a testing cluster – which has a downside where each test can be run one at a time, as GitHub actions run in parallel. When doing complete-backend tests, a mongodb database and a server needs to be instantiated, then requests could be sent. A proposed testing cycle would be: initialising the database, and server, pass server Api calls and then checking the database for changes. This would be a black-box test and hence mocking for Api's should not be required.

Walter will do some work on the checklist time allowing.

Meeting closed at 4:33pm.