Sprint 1 Week2 Poker

Items on the block:
#3, #6, #9, #14, #50, #54

Session scheduled for 8:00pm
Session began at 8:09pm
Attendees: Shang Zhe, Yujian, Parit, Walter, Richard

#3: Task involves implementing the login, registration components. As well as login and registration api calls.

W: 2, S: 3, R:3, P: 3, Y: 4 → size 3

Yujian: Not very invested in hooking up components. Making the components and fields are very easy… experience required

Parit: Aesthetic stuff, lay it out and margin. But hooking up with api Shang Zhe all ready created the api documentation and everything, should be easy to hook things up.

Richard: 'Everything is easy.' Not much you can do. Put some components together and write 'five' lines to make the http requests.

Walter: What if the call fails and rejected.

Richard: You have to display that error message. Also for registration you'd need a password policy.

Walter: Need the character set and all.

Shang Zhe: In addition to setting up the layout, you need to set hooking up to api. Hash passwords for login, as well as lifecycle things such as behaviour upon successful save.

Walter: Lifecycle?

Shang Zhe: How should page navigation occur.

Walter: Redirection to another path should work.

Parit: We could have the same component that will be different whether you are registering or login in.

Walter: I thought it would be simple to implement the login and registration component hookup and all. But one successful login, behaviour may be tricky to implement as from my personal experience our team struggled with Redirection in React.

Richard: We could refresh the whole page, while with the cookie it would bring up the home page.

Shang Zhe: We could do without it.

#6: Tasks involve developing the UI components required for this component. Including the button sockets to connect callbacks to effect on their presentation. The implementation of these callbacks will be handled by #14. Implementation of controller for api calls will be done in #54. Jimp interaction

W: 5, S: 4, P: 6, Y: 4, R: 6

Parit: Card detail is a gateway is everything with the card. We need to know about the card details, from above. Implement api calls for delete card, create card and edit card.

Richard: The create card button will also conjure up this card detail.

Richard: Will there be multiple card detail instances? Do we recycle card detail components or create a new instance each time.

Richard: Activate the card fields as well. Need to respond to two states: edit, read-only. Need to implement edit and delete api.

Shang Zhe: The component will call methods from the controller. This controller will resolve whether to PATCH or PUT.

Walter: Component requires to handle two states. In the editing states need to add fields, delete some fields or modify some fields.

Shang Zhe: Have we prototypes this in our prototype? Adding new fields could recycle the add Tags button as a gateway. It is down to the assignee to implement fields to accommodate for key-value independent change.

Shang Zhe: Looking at the component breakdown there were several things to do. Building card details involved building Fields, ImageFrame (click to select card), CardDetailActions to house the buttons (variable from 3 to 4 buttons depending on state). The middle section is meant to be scrollable, apart from CardDetailActions and the Component Header containing the X.

Yujian: Misunderstood the task. Did not account for the different button actions such as toggle edit and delete being part of this task.

Shang Zhe: Tags are for sprint 2. For this implementation they could be left out.

Shange Zhe: Some decoration or symbol can appear over image to show they can be altered.

W: 5, P: 6, S: 5, Y: 6, R: 5 → size 5

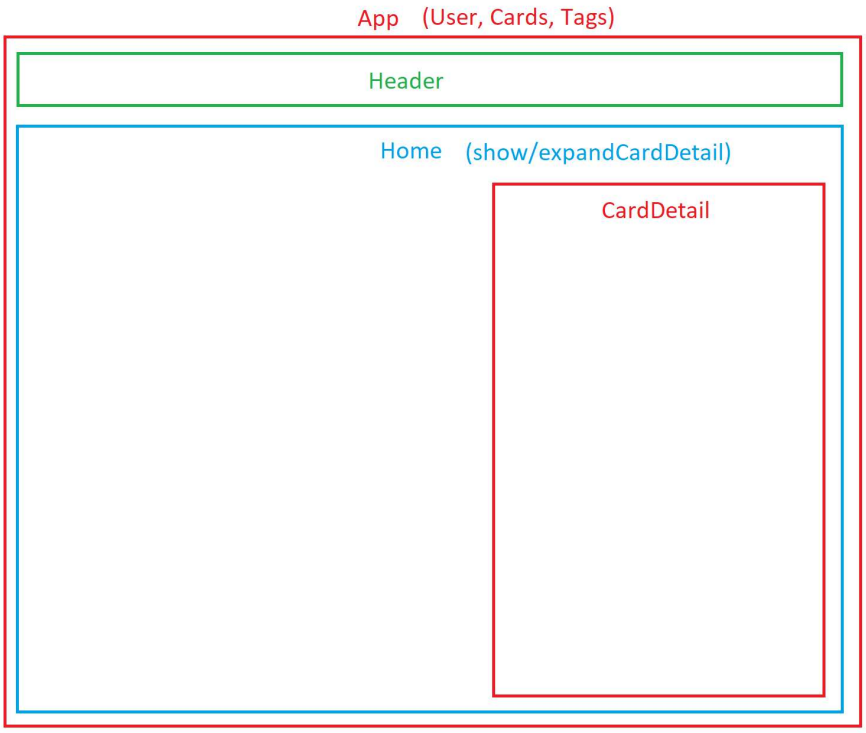
#9: MARKED AS DUPLICATE TO #14

#14: Task involves search to filter out card array for card grid, implement the CardGrid, CardTile Components. Implementing the logic to expose panel and expand / contract panel as described #6 (a). ~~As well as a calling with the Card Controller to retrieve a new card (implementation #54).~~

Shang Zhe (a): Home view has two contexts, one passes the card focus onto the panel. The other provides the state for the 'wideness' of the panel. These implementations will need to be done here.

Parit: For the universal search, does it only handle the result of the search, or the searching query?
Shang Zhe: Searching for the text in the cards data and filtering only appropriate cards. Taking the search string and filtering the cards. This query is present in Global App.

Richard: Is this task involved with storing the results from the /me endpoint?
Shang Zhe: App keeps the context for the user from @/me and this task involves taking in the context for loading. The Home Component itself contains context for showing and expandCardDetail. A stub useApp() exists for now.



W: 6, R: 6, Y: 6, S: 4, P: 5

Walter: Aside from the context. Need to look grid layout to expand and contract.
Shang Zhe: Layout and design can come later, first implement the logic.
Walter: Home needs to trigger reveal card, could be done in CSS?
Shang Zhe: Could try with conditional rendering.

Yujian: Mainly because I'm uncertain how it will communicate with other components. Unsure about how the contexts works. The grid, push in push out could be done easily with CSS. Where we would spend more time on is on communicating with the Card Detail. Second is processing the search String.
Shang Zhe: Look for each card values matching the string.

Walter: Intersection of words or Union.
Richard, Parit, Shang Zhe: Use the intersection.

Richard: It sounds more complicated than card details. There are many interfaces – contexts, props to child components and props looking out to App as well.

Parit: Same with Richard, and Walter. But size 5 because I understood this task better than card details.

Shang Zhe: I could only see some difficulty in filtering cards, otherwise it should be fine.

Comment: Shang Zhe: simulating user and search query, for your local tests, can be done by editing the attributes user and searchquery within App.tsx

W: 3, Y: 3, P: 5, S: 4, R: 4 → size 4

#50: Implement component, including search bar, add card button and profile dropdown. It will receive props from App regarding whether user is logged in. Search bar will set the queryString context on App. ProfileDropdown for this moment will house a button to trigger log out callback (implemented in #54)

Yujian: For the search bar would we like autofill?
Shang Zhe: Not required.
Richard: We have the ability to.
Shang Zhe: I do not think it is useful to auto-complete yet. What can we suggest? Search is live so having suggests is not scope at this time.

Parit: For login out, will it only involve deleting the cookies?
Shang Zhe: Yeah.
Walter: Want to refresh too?
Shang Zhe: I just realised you cannot remove cookie solely using Javascipt. As the cookie is set to HTTP only. Since JS cannot read a HTTP, it cannot delete it. One solution is to implement an endpoint that unconditionally returns a clear cookie response (a size one enhancement).
Walter: Will it then refresh?
Shang Zhe: Could reset all contexts.
Walter: Then refresh or redirect?
Shang Zhe: that would be down to #54.

W: 3, Y: 4, P: 3, R: 3, S: 2

Yujian: Two notable points. The header itself has the add card function. I am unsure about how that's going to be done, as well as how the logout would be achieved.

Walter: Add card button will contact the controller for a new card. Will this be sent to the context?
Shang Zhe: I need to create an upstream channel to have access to the home context if there is one. That is the tricky part because the home component is sister to the header. I will have to move show card detail up to App and expand can remain in the context for Home. This will allow the constructed cards to be set in the App context.
Walter: And I assume that the controller will not need to be told what is happening from the UI.
Shang Zhe: No it will be able to infer itself.

Walter: The profile dropdown could just be a FluentUI Dropdown component.
Shang Zhe: A split button could also work.

Parit: Search box changes searchQuery. Log out call a function. Add card will notify the card creator.

Richard: It's not that difficult. Same size as login.

Shang Zhe: It's really a textbox and two buttons.
Walter: Will FluentUI offer icons for the buttons?
Shang Zhe: Yeah.

→ Size 3

#54: Application lifecycle involves implementing all Card controller functions. Connecting the application to all the endpoints on the backend. This task will be responsible for updating commit() and delete() for the card controllers – with touch the backend. It also is responsible for managing login / logout, sending initial @/me request, registration. Application context and interactions with the api will be implemented here.

Shang Zhe: It also involves decoding JSON results into forms convenient for Components to use. Uploading the image will be done here, PUT and PATCH.

Walter: Which issue deals with GET image?

Shang Zhe: Done in #14 & #6. Since the card tiles will be implement there. It would be good to let these make their own Api calls for the images. GET into image endpoint. These can be done implicitly by the browser through <img src> or through FluentUI Image Component.

W: 6, R: 6, P: 8, Y: 9, S: 7

Yujian: App lifecycle. Absolutely no idea on how to start writing. It involves everything and has to be the backbone of everything. Debugging and testing will be a big doing as well.

Parit: It got to be the centre of everything. Make sure the functions work properly when other parts of the application calls it. It has a lot of responsibilities. Debugging could be a pain.

Shang Zhe: I'm playing it safe. Sending the network request is not the hard part, but decoding and representing it correctly such that React does accept it. This is unfamiliar territory. However, I am confident that the clean interfaces defined will keep the confusion gap low.

Walter: Among the responsibilities. The api calls will need to affect contexts and trigger redirections if necessary. From my experience redirection has to be done carefully lest state be lost and components unaltered.

Richard: Even though there's the amount of work to be done. Because of how interconnected it is you need to have a very strong high level understanding of the component tree. I feel that it's not much coding, but that understanding of the system is required.

W: 7, S: 6, R: 7, P: 7, Y: 7 → size 7

#55: Implement logout endpoint that unconditionally removes token cookie from client.

W: 1, R: 1, Y: 1, P: 1, S: 1 → size 1