Team PorkBelly
Sprint 2
Size Poker
15th September, 2021

Held at 8:05 pm
Participants: Walter, Shang Zhe, Yujian, Parit, Richard

Issues on the block: #10, #56, #64, #65, #76, #77, #78

#10 – Add card tags feature

Adding, editing and deleting tags functionality to endpoint /api/tags for putting new tags for a user, patching the detail of a tag, and specifying the deletion of a tag. Implement controller callbacks in the front-end to make the appropriate requests.

Implementing the tags picker UI in card detail to assign / run these CRUD connectors, tags picker in the Home component to filter cards based on the tags selected, filtering cards on AND basis. Front-end work also involves implementing the logic for sifting cards for qualifying cards. Finally, to reflect these selections and allow users to remove their tag filter, the search bar needs to include the selected tags with controls to remove them from the query.

#10 will be split.

Endpoint:
A: PUT/ PATCH/ DELETE

Front end:
B: Extend AppContext to to support tag filter – definition.
B1: Definition of the Tags Component's Prop.

C: Controller – implementation for Tag methods
D: Create a Tag Component, display of name, colour, button to dissociate, etc…
E: Tags picker UI

F: Extend FilterCard in Home.tsx to accept changes to the tag filter.
F1: Scrolling presentation for tagList.

G: Implement logic to show tags in search bar with buttons to remove from filter

A: PUT/ PATCH/ DELETE
Endpoint implement ion for Tags CRUD. Creating a tag and tying to a user, updating its details, deleting a tag will need to remove itself from all cards to prevent dangling references.

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

Yujian: While the tasks are similar to card CRUD tasks, they were a size of 3 individually. I would give it a more because of you need to dissociate tags from cards on deletion and from users.

Parit: 4 as a combination of 3 paths.

Walter: To find which cards to update, you'll need to search for existence in tags field for each card. Mongoose better have a service for this. Shang Zhe stated that as mongoDB supported this, Mongoose should do so.

Richard: With Tags endpoint there was less sanitation needed for the fields. The code may be shorter and the implementation should be simple. The implementation of these endpoints are very similar so consequent implementations are straightforward.

Shang Zhe: It does not involve any research. We know exactly what needs to be done.

AMENDMENT: Defining RESTful Api.

W: 3, S: 2, R:3, Y: 3, P:4 – Size 3 → Shang Zhe

B: Extend AppContext to to support tag filter – definition.
Implement the definition of the tag for the front-end. Extend AppContext to support a collection of target tags for the application to filter for qualifying cards. This extension to context will be used by required components to effect on the query or display the query (namely, searchBar, cardGrid, cardDetail).

Definitions are required for the other front-end issues.

W: 2, R: 1, S: 3, Y: 2, P: 3

Shang Zhe: Called 3 because it cannot be done alone, requires the presence of another party.
Walter: Please elaborate.
Shang Zhe: Due to the importance the dependence of other tasks on it, it's not advisable to be done alone.
Yujian: Is that more of a time estimation or size?
Shang Zhe: More size as it's a back and forth thing.
Walter: In favour of this.
Shang Zhe: We should be doing in a call in close future.

Parit: Required understanding of all the other functions.
Shang Zhe: Yes, it involves foresight.

Walter: You need to define the Tag interface. Then you need to declare methods to update the filter query and the field of the filter query.
Shang Zhe: Tag interface will contain methods for its control. These controls will be implemented in the controller tasks.

Yujian: I had not considered the communication of components in the front-end. Not much to add.

Richard: Did not consider the communications required. While it requires experience, my size estimation was tied to time – where it would only involve implementing the interfaces. However, now that it was recommended as a group thing, it should not hold the role of a task (and hence size should not apply) as it becomes a team meeting.
Shang Zhe: Still have a size to communicate, how much there is to discuss.

W: 3, R: 3, Y: 3, S:, P: 3 → Size 3 task :: set a date for meeting → Shang Zhe

B1→ Size 2, need to agree on what is required to be shared. → Everyone

C: Controller – implementation for Tag methods.
Implementation for methods on the Tag interface, and AppContext extentions (on app.tsx). Implement CRUD calls, stage, commit, or something. Dependent on B for signatures.

W: 4, S: 5, Y: 4, R: 5, P:3

Richard: there's a learning curve to implementing the task. You'll need to keep the coding style and standards consistent. You'll also need to make the proper tests. It's a lot of lines of code.

Shang Zhe: It requires a degree of familiarity of how and why current implementations of controllers are written this way. It's a slight deviation from 'conventional' practice due to the peculiarities of JS. It also depends on foresight and experience to do the job.

Walter: Foresight like extensibility?

Shang Zhe: Experience and knowledge of the input/ output, limitations and capabilities of JavaScript, React lifecycle – part of why it's in one file. And to have a plan for creating the executions.

Walter: Implement for calling the appropriate endpoints, validation of fields, parsing the response. Presenting the search query into context.

Shang Zhe: Effecting of the filter query will be carried out in the required components.

Yujian: There is a sizeable chunk of code to be done. Similar structure to Card.ts and .tsx so there are similar code logic. Less fields in cards too. I did not consider the foresight aspect.

Walter: Is the foresight just on the api calls.

Shang Zhe: It's more than that. Requires implementation of methods to align with the react lifecycle.

Richard: What about how devs will use the methods?

Shang Zhe: That will be set in part B alongside the guideline of use.

Parit: Similarity to controllers/card.ts but it may not be the case.

W: 5, R: 5, S: 5, Y: 5, P: 5 → Size 5 → Shang Zhe

D: Create a Tag Component, display of name, colour, button to dissociate, etc…
Tag React Component that displays presents a tag. A button for an ‘x’ callback. Needs to support ‘read-only.’ Component will be used for CardPanel, and SearchBar to display tags.

Home page tagList might include this component.

W: 2, S: 1, R: 1, Y: 2, P: 4

Parit: Functionality of the button needs to be configurable to the context the component finds themselves in.
    Walter: The implementation for these callbacks will be implemented in the parents. For searchBar it would be implemented in the searchBar and callback for card dissociation on cardPicker.

Yujian: Because the component would be used in many other places, it would need to be adaptable to achieve the different purposes for the job.

Walter: Tag component as a glass box – only purpose is to display and call a callback of the tag.

Richard: It’s a box inside a box with colour, text and two buttons.

Shang Zhe: It’s a presentation of the label and the button. No sweat.

W: 1, S: 1, R: 1, Y: 2, P: 2 → size 1 → Parit

E: Tags picker UI
Task focused on creating the component for the purpose of extending CardDetail.
Either an extension of CardDetail or a separate component (recommended for modularity), as it so far is only being used in cardDetail Panel. The component will need to be made nonetheless. make the callback to remove a tag from a card (for cardDetail), add a tag to a card, call controllers functions to edit a tag, create a tag, delete a tag.

Implement the UI to display the tags available, controls to bring up a tag edit panel (may be implement as another component), as well as UI to display the fields to edit the card.

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

Parit: pop up menu windows needed, color picker needed, create tag button and functionality needed

Yujian: The things to be done. Two levels of pop-up not as easy to implement. Also would like to make it modular.

Richard: It would be more productive to make it modular.

Walter: Two layers of popups. You need to display tags. Need to have state to show expansion and collapse modes.

Richard: Same size as CardDetail, or so it appears. My vision of it is two components. You need to create circles with colours in them. You need tags, put them in the correct structure. Putting things in the correct structure.

Shang Zhe: It’s not that much work. Three lists and two pop-ups. The component should not be long. Making circular buttons would be left to visual styling. Functional first.

Props defined in B (#81)

W: 5, R: 5, Y: 5, P: 6, S: 5 → size 5 → Walter

F: Extend FilterCard in Home.tsx to accept changes to the tag filter.
W: 2, S: 1, R: 1, Y: 2, P:1

Walter: Get the tags. Filter on an intersection of the tags obtained from the context.

Yujian: Straightforward, but what is the interaction with keyword search.
Shang Zhe: The query is at an intersection.

Shang Zhe: Get the tags from context. Filter is a function call. The foundation is there.

Richard: Quite possibility a one line job

Parit: Ditto to Shang Zhe and Walter

Closed → Size 1 → Yujian

F1: Scrolling presentation for tagList.
Display the tags a la Pixiv.net. Upon clicking the tags, it would send a callback to Context to accept the tag into the tag filter (part of B).

W: 2, R: 1, S: 1, Y: 1, P: 2

Parit: append tag id/ name to the app context, --check for duplicated tag--
Walter: Duplication check would be done by AppContext methods. This component will make the callback to the context.

Walter: It's a list of Tags to display, each with an on click property:

Yujian: Ditto to Walter.

Richard: It's a list or horizontal stack that is written by someone else. Horizontal scroll-able stack.
Walter: Are the buttons included in FluentUI?
Shang Zhe: Those will need to be attached. Making callbacks to scroll the stack by a page. The scrolling function comes from HTML – scrollby();

Shang Zhe: Ditto.

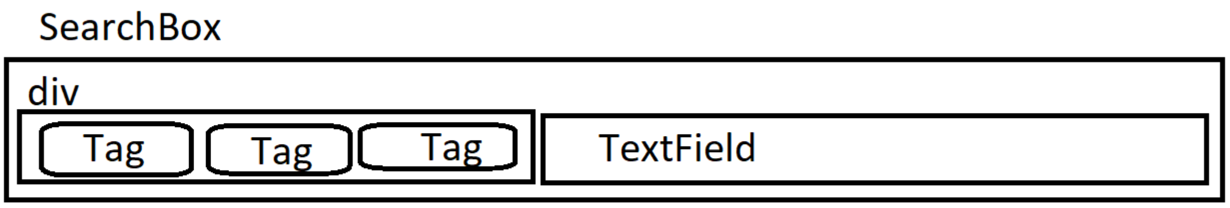
Closed → size 1 → Yujian

G: Implement logic to show tags in search bar with buttons to remove from filter
Display tag filter as tags inside the search bar to the left of the text input. Tags would make the callback to remove itself from the search query when the ‘x’ button is clicked (Defined in B).

W: 1, R: 2, S: 2, Y: 2, P: 3

Parit: Displaying tags inside search box may be tricky.

Shang Zhe: You add an invisible div inside the searchBox to contain the tags, next to a textField to insert the text. The containing searchBox can be made to scroll if there is an overflow.



Richard: A box inside a dive with textfield. If backspace to delete a tag, it could be a bit more complex.

Shang Zhe: If there is not a requirement. No need to implement.

Shang Zhe: I’m automatically expecting it to be finicky. The act of putting textfields inside other things, which are containers that may contain large margins. This could lead to unexpected overflow behaviours.

Yujian: Would this be a problem for CSS to solve?

Shang Zhe: I’ll need to research how overflows are dealt.

Richard: Things showing up where they should not be is broken, not ugly. So that issue is something this task could be tested with.

Yujian: Ditto Richard.

Walter: Did not consider overflow.

Closed → size 2 → Parit

#56 Visual Styles
Everything to do with presentation. All at once.

Making the website visual styles to match the prototype v2.3. Not withstanding measurements as the prototypes as yet to be accurately measured.

What needs to be styled:

- Header
    - Brand Logo
    - Search Box
        - Tag Component
    - Add Card Button
    - User Button
        - MenuDropdown
        - Logout Button

- Login View
    - Username, Password Fields
    - Page Background
    - Buttons and links

- Home View
    - Tag List
        - Tag Components
        - Scroll Buttons
    - CardGrid
        - Card Components
            - Image
            - Labels
            - Focus Decorator
    - CardDetails
        - Close Button
        - Image
        - Associated Tags list
            - Tag Picker
            - Tag Editor
            - Add Tag Button
        - CardFields
        - Note box
        - CardDetailAction
            - Expand Button
            - Edit Button
            - Commit Button
            - Cancel Button
            - Delete Button

W: 8, Y: 7, S: 6, R: 10, P: 6

Richard: Not hard but a lot of work. More of a time estimation than a difficulty.

Walter: Not only is it to conform the prototype. Alignment can be a bit of trouble. Will need to scour the entire application. Has to wait on components that need to be implemented.

Richard: We can start with the things ready?

Shang Zhe: When styling close to construction sites, you’ll need to make move things around once the buildings are finished. Intended that this would commence after the functional tasks are completed.

Yujian: Along with Pep and Richard where there are a lot of small things to do. I do know that some parts will need more attention than others, like Tag Picker over buttons. Because we have a prototype it is much easier to make something up with a reference.

Parit: not hard but a lot of work

Shang Zhe: Just about the most finicky task. You’ll have to battle HTML5 to get the alignment you desire.

PREQ: Measurements

W: 7, R: 7, S: 7, P: 7, Y: 7 → size 7 → Shang Zhe & Yujian

#76: Change API to send hash instead of hasImage
The team noticed that the image components kept refreshing. It currently spams the back-end for image GETs. This result is deterministic on the card's image. It allows the browser to cache the image once it is fetched. This involves changing the backend to set caching settings for image update.

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

Parit: Unfamiliar with the task.

Yujian: To quote Shang Zhe it is going to be finicky. Even changing this would not yield optimal results and may require extensive troubleshooting.

Walter: If the target is to set the image url to be deterministic. Need to change the GET image to set a cache. Url needs to be returned from the server containing a deterministic hash of the image for the browser can use the created url to trigger a new card get.

Shang Zhe: We could instead return an optional hash to compose the url on the front-end.

Shang Zhe: Good luck reading through App.tsx to update the controllers. It really is just that. Modification of the api in the docs branch. Extend the card schema to hold the image hash. Modify the types in the shared package. Modify card PUT, PATCH and GET me to set / return hash for images. Modify components to create the url with the hash to pass into the img src.

Walter: Do we want to create a middleware to make the hash?

Shang Zhe: Not required, this needs to be done in a transaction on the db.

Richard: Ditto Shang Zhe, except I'll tread through with a Ctrl+F.

Closed → size 3 → Richard

#77: Scroll card into view when selected to view detail
When you expand the card detail you need to maintain focus on the card that has been selected. You need to use a ref and an useEffect hook. Ref to access the card component for scrollInToview(), useEffect when the selected card gets changed. Specifying refs inside of props is deprecated but using useRef() hooks is still allowed. This can be held in cardTile to obtain its own HTML element.

W: 3, S: 1, R: 3, Y: 2, P: 2

Walter: You need each card tile to have a ref and have the effect hook to call scrolling function.

Richard: Not really that familiar with React refs. I thought it could be a few lines but unfamiliar with it.

Parit: just add a line to call ref + effect hook to each card (extension for the Card Component).

Yujian: Ditto Walter + Pep

Shang Zhe: Ditto

W: 2, S: 1, R: 2, P: 2, Y: 2 → Size 2 → Richard

#78: Process image in parallel
JIMP will have a hard time to load and process the image, freezing the UI thread and making the website look unresponsive. Using WebWorker Api to use a separate thread for this task frees up processing for the UI. Need to lookup WebWorker Api for threads.

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

Parit: Not familiar with the task.

Richard: Looks pretty easy, unsure of the problems that could arise.

Yujian: Ditto Richard. Documentation on MDN doesn’t seem overly long.
    Richard: How are messages passed between threads? A learning curve to this.

Walter: Race conditions between committing and image upload.
    Shang Zhe: If you click commit before the image is posted then it would register as no changes, take up changes up to the image change and also subsequent modifications to the fields.
    Richard: Could disable the commit button, while the worker is working.

W: 3, Y: 3, R: 4, S: 3, P: 3 → Size 3 → Richard