**Functionalities**:

------------------------------------

Instead of using a separate timer for all the items in the queue. The top item in the queue should be considered special with actual timer, rest of the items will have a fake timer and it’s value will be dependent on the top item in the queue.

**Queues**:

Main queue: only queue which supports all the operations

Completed queue: queue to store the list of completed items

Cancel queue: queue to store the list of items cancelled.

-------------------

**Operations on the queue item**:

**Cancel**: cancelItem()

Cancel and remove the item from the queue, remaining items in the queue should get adjusted on their own.

Item placed in the cancelled queue list

**Skip**: skipItem()

Skip operation applicable only for the item in the top of the queue, if skipped the next item in the queue will become the top item.

**Complete**: completeItem()

Item has completed its lifecycle in the queue and hence removed from the queue and placed in the completed queue list

**Timer component**:

If Timer component is completed for the top item in the queue,

Notification is sent to the queue manager, where the queue manager can either complete it or mark as delayed.

Completed: item removed from the queue and added to the completed queue.

Delayed: item continues in the queue with a original timer 0 and a new delay timer which does counting up

Problem: how should the remaining queue items show the timer?.

**Form:**

Form can add items to the queue with a job date. On save the items added to the queue are saved with the date.

On the particular date the queue for that day is loaded, queue manager can click start when the timer for the queues are ready to begin.

Even while the queue is active. The queue manager can add items to the end of the queue.

**Components in the queue item**:

**Timer**: Timer indicates the estimated time in which the item will remain in the queue.

ID: unique id to identify each queue item. Validation should be added to check if item is already in the queue.

**Name**: Name of the queue item.

**Description**: can be any plain text used to identify why the item is in the queue. This should have an expand option, only if clicked this needs to be viewed to the actual user.

**Token No**: Another important field is the token no in the queue, when an item is added to the queue, the token number is the last token number + 1.

**Waiting number**: waiting number is the index of the item in the queue, token number will not change, but the waiting number will change.

Try to incorporate the timer function only at the end, since it can get complicated. Make sure all the other queue functionalities are ready before that. Timer component should be implemented as a standalone so that it can be reused in any other projects as well.

Additional functionality:

Try to implement using serverless function:

Save: should add the items for today in the queue to the database

Day completed: save the details of completed as well as cancelled details to the database

Load queue: should load the details from the database for today.

Also the queue details based on the date in calendar should be made visible on clicking the date

ALGORITHM

OPERATIONS ON THE QUEUE

cancelItem(item):

cancelQueue.push(item)

queue = queue.filter(queueitem.id != item.id)

skipItem(item)

swap with the next element

tmp = item[0]

item[0] = item[1]

item[1] = tmp

completeItem:

completeQueue.push(item)

queue = queue.filter(queue.id != item.id)

TDD

Test cases

Case 1:

Add items to the queue

Case 2: Mark item from queue as completed

Should get removed from current queue and placed in completed queue

Case 3: Skip item from queue

Should be pushed one position down

Case 4: Cancel item

Should be moved from queue to cancelled queue

Bugs:

Token id set back to 1 when queue list is empty

To do: 26th march 2022

1. New token generation function
2. Refactor the code, take queue operations outside and perform set state alone inside the component
3. Make use of context hook to avoid passing everything as props
4. Validation on unique ID
5. Use serverless functions to store the data to the database.

When to store the data??

Schema definition?

How many documents?