Files Expired From Azure Storage Queues

QA did a mass apply that got out of hand, it ended up adding so many files to the file processing queues, that we had many files queued in file processing expire out of the queue after 7 days because file processing did not process them fast enough. Luckily, the state of each file that is being processed is stored in Cosmos DB so we added some functions that, given a specific time range, will go and requeue items that did not finish processing in that time period.

Steps to Do This

1. Use the query at the bottom of this page to see how many items were potentially dropped from the queue in a given time period.
   - Dates should be in UTC and formatted like this => '2024-08-05T00:00:00.000000Z'
   - When you run the http function in the next step, this is how many items will be put in the `processrequeuedexpireditemsqueue` queue in the `revvertextextractprocus` storage account.
   - If there are more than 200k items queued it will likely fail to queue all items successfully (which is why you are doing this step in the first place, to validate the expected number of items actually get queued), so you may need to narrow your time window and run this process multiple times if there are more than 200k items.
2. Make sure the `ProcessRequeuedExpiredItems` function in the `revver-fileProcessingManager` is disabled.
3. Run the http triggered `RequeueProcessingExpiredItemsFunction` function in the `revver-fileProcessingManager` function app with a request body like the following.
   -
     ```
     {
       "StartTime": "2024-07-31T00:00:00.000000Z",
       "EndTime": "2024-08-03T00:00:00.000000Z",
       "AccountIdsToExclude": [6878]
     }
     ```
   - The `AccountIdsToExclude` property allows you to not requeue items from a specific account (it's possible you just want this to be an empty list, in the first case, we didn't want to requeue items in the QA account)
   - The http triggered function will likely appear to time out, but it should continues processing regardless, but you will validate that the expected number of items were queued in the next step for this reason.

4. Ensure the expected number of items are placed in the `processrequeuedexpireditemsqueue` queue in the `revvertextextractprocus` storage account
   - you validate this against the result from you query in step 1
   - if it didn't queue the expected number, clear the queue, and narrow your time window
5. disable the InitiateOCRExtraction function in the revver-ocrProcessing function app
6. disable the InitiateTextExtraction function in the revver-textExtractionProcessing function app
7. enable the ProcessRequeuedExpiredItems function in the revver-fileProcessingManager function app
8. monitor the `processrequeuedexpireditemsqueue` queue in the `revvertextextractprocus` storage account and wait for the queue to be empty
   - as items are processed in this queue, messages will be added to the initiate ocr and text extraction queues
   - not all items will be requeued to either queue, some items we cannot determine were not processed or not until this step, so often many of the items will not be requeued.
9. disable the ProcessRequeuedExpiredItems function in the revver-fileProcessingManager function app
10. enable the InitiateOCRExtraction function in the revver-ocrProcessing function app
11. enablethe InitiateTextExtraction function in the revver-textExtractionProcessing function app

Query for expected number of items to be queued (run in cosmos db file processing items container data explorer)

```
SELECT VALUE COUNT(1) FROM c
WHERE
c.AccountID NOT IN (6878)
AND (
    (c.Timestamp > '{start time here}' AND c.Timestamp < '{end time here}')
    OR
    (c.Pipeline.TextExtractionProcessing != null AND c.Pipeline.TextExtractionProcessing.Date != null and
(c.Pipeline.TextExtractionProcessing.Date > '{start time here}' and c.Pipeline.TextExtractionProcessing.Date <
'{end time here}')
    )
    OR
    (c.Pipeline.OCRProcessing != null AND c.Pipeline.OCRProcessing.Date != null and
(c.Pipeline.OCRProcessing.Date > '{start time here}' and c.Pipeline.OCRProcessing.Date < '{end time here}')
    )
)
AND (
    (
```

```
      c.Pipeline.TextExtractionProcessing != null AND (c.Pipeline.TextExtractionProcessing.Status = 'InProgress'
 OR c.Pipeline.TextExtractionProcessing.Status = 'NotStarted')
    )
  OR
  (
    (c.Pipeline.TextExtractionProcessing != null AND c.Pipeline.OCRProcessing != null)
    AND
    (
        c.Pipeline.TextExtractionProcessing.Status = 'NotApplicable'
        OR
        c.Pipeline.TextExtractionProcessing.Status = 'Complete'
        OR
        c.Pipeline.TextExtractionProcessing.Status = 'Error'
    )
    AND
    (
        c.Pipeline.OCRProcessing.Status = 'InProgress'
        OR
        c.Pipeline.OCRProcessing.Status = 'NotStarted'
    )
  )
)
```