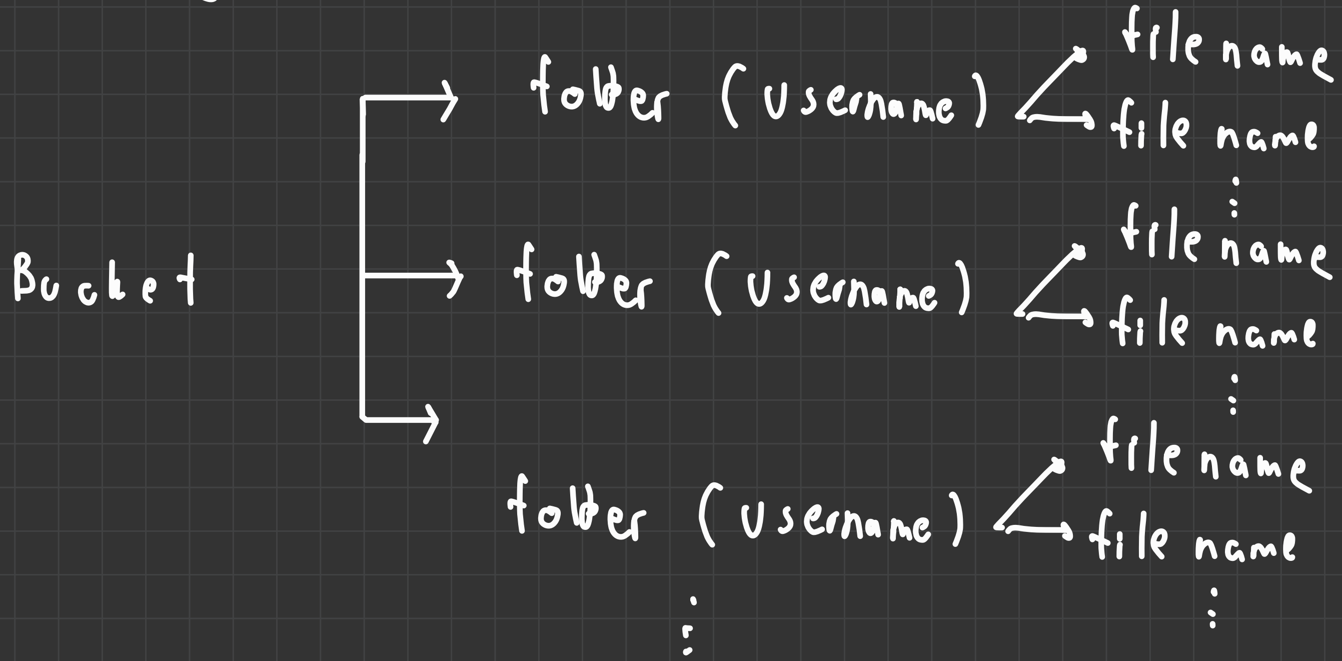
S3 design



Bucket
folder (username)
file name
file name
⋮
folder (username)
file name
file name
⋮
folder (username)
file name
file name
⋮
⋮

Dynamodb design

myDropBoxUsers

Partition key	$\rightarrow$ username
column	$\rightarrow$ password

myDropboxItemOwnerships

Partition key	$\rightarrow$ username
Range key	$\rightarrow$ filename
secondary index	$\rightarrow$ lastModifiedDate

my Drop box SharePairs

- Partition key → sharee (Username of who get file permission from sharer)
- Range key → sharer (Username of who give file permission)
- Column → sharePairkey

For query here

my Drop box PairToItems

- Partition key → sharePair key
- Range key → file name

README

[Client]

I decided to always get response message if there are error so I reduce checking in client server to just check if there are message in response

[Server]

In my lambda I combine all action into one function which got a flow like this

```
username and password are needed to contain in body
- Check if contain action, username and password or not
    - [RETURN] if there are not exist return message to told that

- [dynamodb] query user from user table

[newuser]
    - Check from dynamodb if there are username exist or not
        - [RETURN] if exist return  return message to told that
    - [RETURN] empty object

- now another action are need to have username and got correct password
-  Check if user exist or not
    - [RETURN] if user are not exist return message to told that
- Check if password match
    - [RETURN] if password not match return message to told that

[login]
    - [RETURN]Now if we come to this step the username are found and password is match
    so we just send back empty object

[view]
    # we need to get all filenames from 2 ways, own item and share item
    - find all user own file from itemOwnership table
    - find all pair that user is getting shared from sharePair table
    - iterate through all sharePairs match
        - using sharePairKey to get all pairToItems range key (filename)
        - I collect all filename into toQueryFilenames list
        - now there are missing variable which is latestModifiedTime
        # Now we know sharer from sharePair and toQueryFilenames from pairToItem
        - Query from itemOwnership with sharer username and filter it with
        toQueryFilenames
    - Through all iterate we got all file data we needed
    [RETURN] return all file in format { fileLists: list of all files }

# Now the rest action we needed to include filename in so if there are no filename we would return
message about that
    - [RETURN] if there are no filename

- We rename it to s3_filename which in folder of username contain in body
[put]
    - set lastModifiedDate and create newItem in format {username, filename.
    lastModifiedDate}
    - put new item in itemOwnership table and generate presigned url for upload into s3

# Now the rest action we need to check if file are exist or not
- set res = {}
- query from fileOwnership table to check if file exist and set found as boolean to tell about it
```

```
[get]
    - if exist set and sharer not in body generate get presigned url for get from s3 and set to res url
    - if sharer in body find from sharePair from below flow
        - find that sharePair exist if not set res message to told that
        - else get sharePairKey from sharePair
            - find from sharePairKey if the filename and sharePairKey query match in the pairToItem table
            - if match set presigned url for get from s3 and set to res url
            - else set message to told that file not found
    [RETURN] res

[RETURN] Now the last action is share if found is False return to told that file not found

[share]
    - check if there are exist sharee username from user table
    - [RETURN] if sharee are not exist and set message to told that
    # we already ensure that we got filename in this user
    # we already ensure username does not contain '/' to make sure sharePairKey not duplicate
    - setSharePairKey in format sharee + '/' + sharer
    - create new sharePair to it table with { sharee, sharer: username, sharePairKey }
    - create new pairToItem to it table with { sharePairKey, filename }
    [RETURN] message that are success note that in share we doesn't have to have any other process in client so we can send success message in response

[RETURN] this mean command is not valid and set response message to told about it
```

[HOWTO]
Request api to
https://kpyc78zcuf.execute-api.ap-southeast-1.amazonaws.com/default/dropboxFunction

With following body depend on each case

[newuser]
```
{
	'username' : username of new user,
	'password': password of new user
}
```
You would get response in format {}

[login]
```
{
	'username' : username of user,
	'password': password of  user
}
```
You would get response in format {}

[view]
```
{
	'username' : username of new user,
	'password': password of new user
}
```
You would get response in format
```
{
	fileLists: [
			{
				'filename': name of that file,
				'lastModifiedDate': last modified date of that file,
				'username': owner username of that file
			}
			…
		]
}
```

[put]
```
{
	'username' : username of new user,
	'password': password of new user,
	'filename': 'filename to upload
}
```
# with put we will get presigned url to upload to s3 with limited time as response in format
```
{
	url: 'url to upload',
	fields: 'header for set into the upload request
}
```

and you need to send request to url with format

```
{
        data: fields,
        files: ‘file for upload’
}

[get]
{
        ‘username’ : username of new user,
        ‘password’: password of new user,
        ‘filename’: ‘filename to upload,
        [optional] ‘sharer’ : sharer username if we get file from other user
}
response in format
{
        url: ‘url to download’,
        fields: ‘header for set into the upload request
}
and  you need to send request to url with format
{
        data: fields,
}

[share]
{
        ‘username’ : username of new user,
        ‘password’: password of new user,
        ‘filename’: ‘filename to upload,
        ‘sharee’ : sharee username to share file to
}
response in format
{
        ‘message’: complete message
}

All other except share would get
{
        ‘message’: error message
}
If the action is error
```