MYDROPBOX APP

Rawit Lertluksanaporn 6570201021

AGENDA

Mydropbox app design

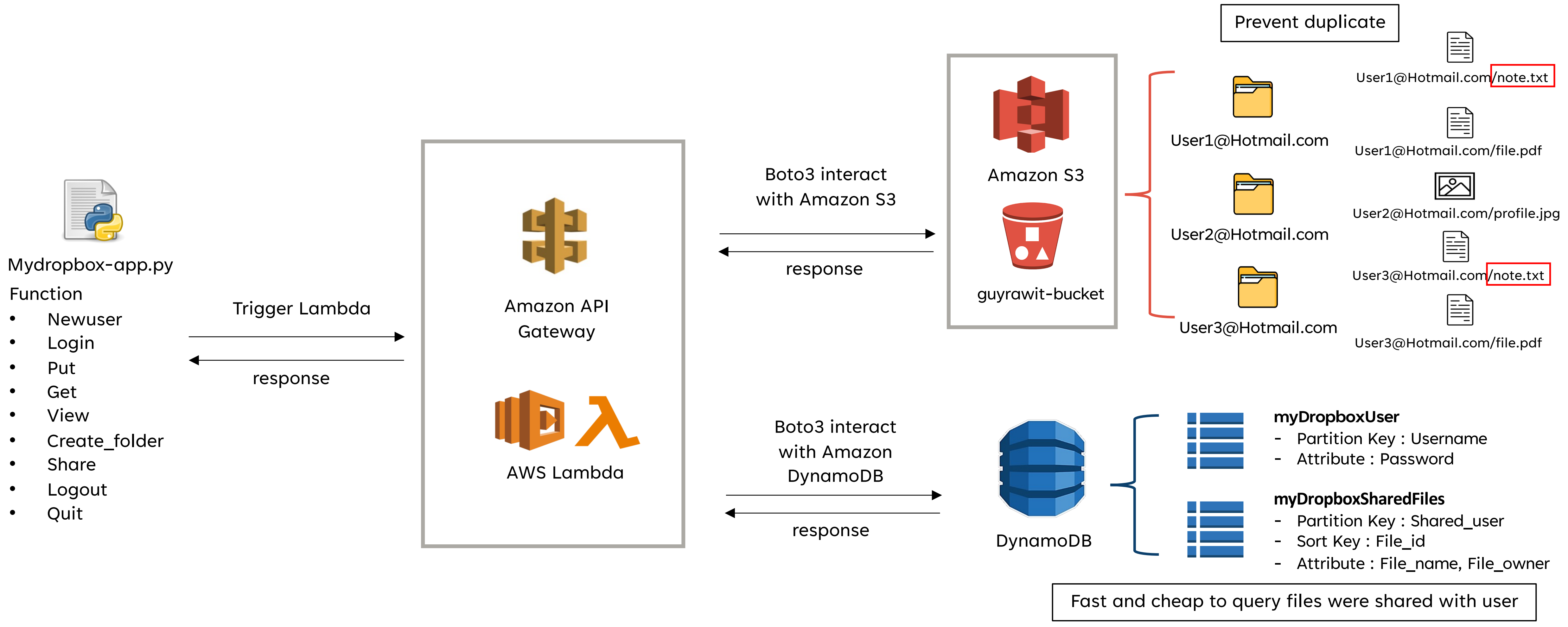
S3 storage design

DynamoDB design

README

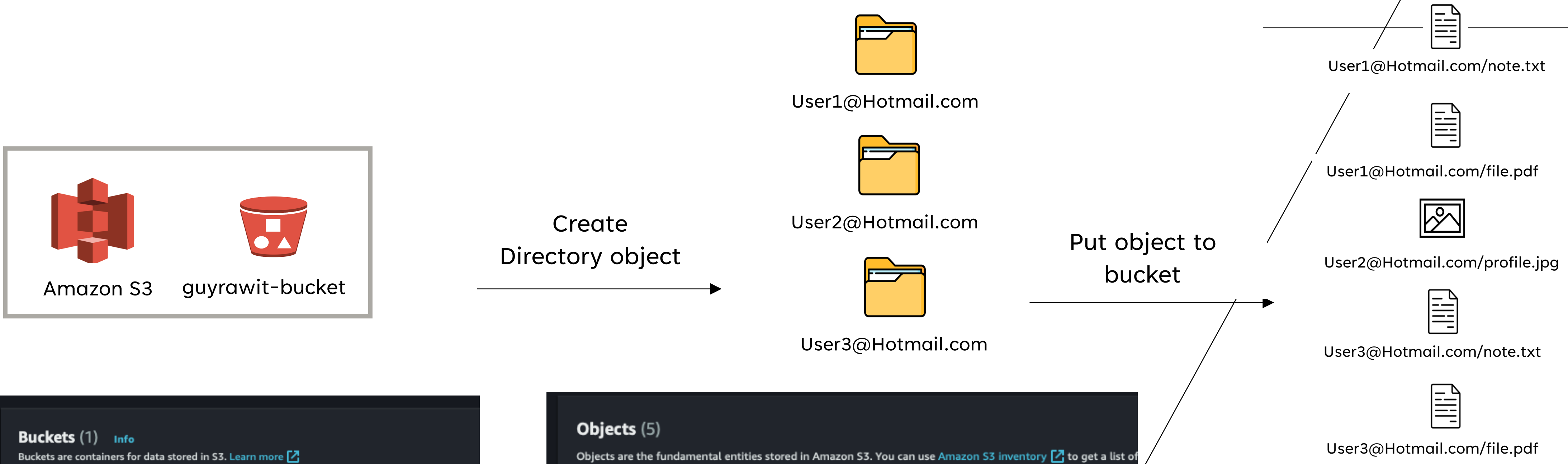
HOWTO

MYDROPBOX APP DESIGN

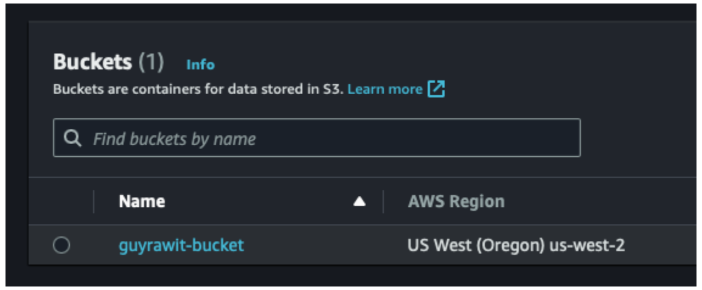

Mydropbox-app.py
Function
• Newuser
• Login
• Put
• Get
• View
• Create_folder
• Share
• Logout
• Quit
Trigger Lambda
response
Amazon API Gateway
AWS Lambda
Boto3 interact with Amazon S3
response
Amazon S3
guyrawit-bucket
Prevent duplicate
User1@Hotmail.com
User2@Hotmail.com
User3@Hotmail.com
User1@Hotmail.com/note.txt
User1@Hotmail.com/file.pdf
User2@Hotmail.com/profile.jpg
User3@Hotmail.com/note.txt
User3@Hotmail.com/file.pdf
Boto3 interact with Amazon DynamoDB
response
DynamoDB
myDropboxUser
- Partition Key : Username
- Attribute : Password
myDropboxSharedFiles
- Partition Key : Shared_user
- Sort Key : File_id
- Attribute : File_name, File_owner
Fast and cheap to query files were shared with user

S3 DESIGN

Prevent duplicate objects name
By using username as prefix



Amazon S3
guyrawit-bucket
Create
Directory object
User1@Hotmail.com
User2@Hotmail.com
User3@Hotmail.com
Put object to
bucket
User1@Hotmail.com/note.txt
User1@Hotmail.com/file.pdf
User2@Hotmail.com/profile.jpg
User3@Hotmail.com/note.txt
User3@Hotmail.com/file.pdf

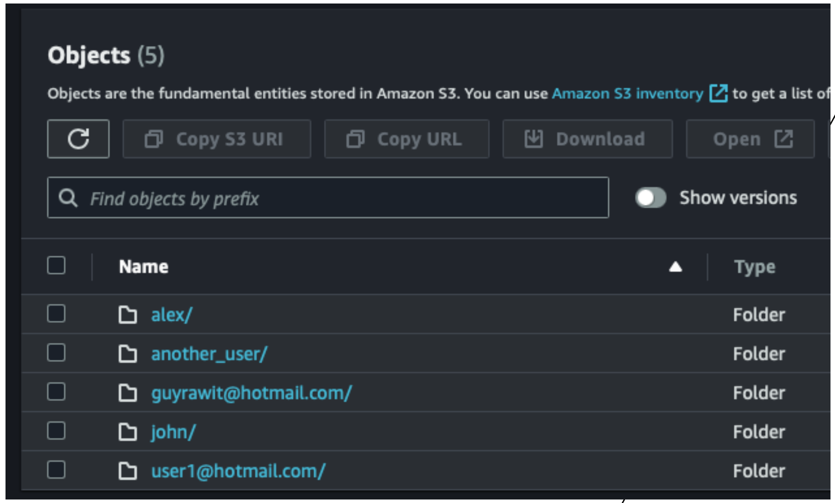


Buckets (1) Info

Buckets are containers for data stored in S3. Learn more

Find buckets by name

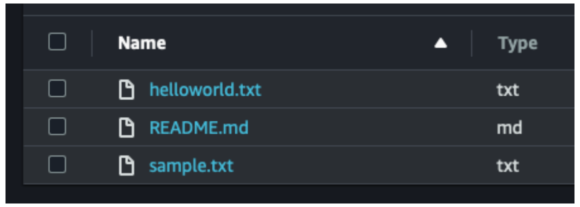
Name	AWS Region
guyrawit-bucket	US West (Oregon) us-west-2



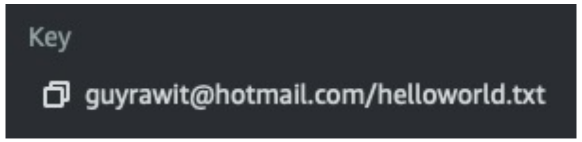
Objects (5)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of

Name	Type
alex/	Folder
another_user/	Folder
guyrawit@hotmail.com/	Folder
john/	Folder
user1@hotmail.com/	Folder



Name	Type
helloworld.txt	txt
README.md	md
sample.txt	txt

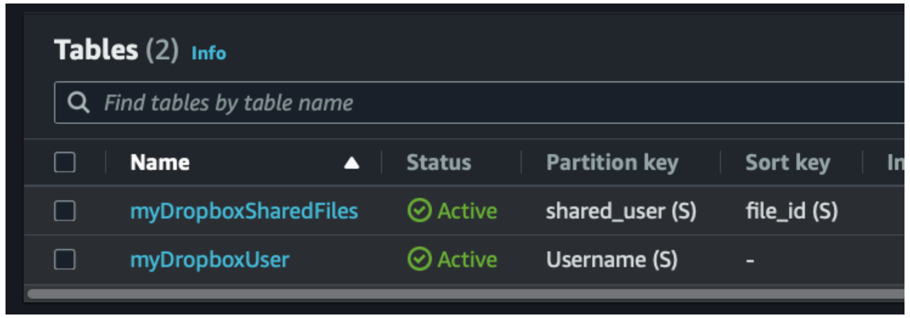


Key

guyrawit@hotmail.com/helloworld.txt

DYNAMODB DESIGN

DynamoDB



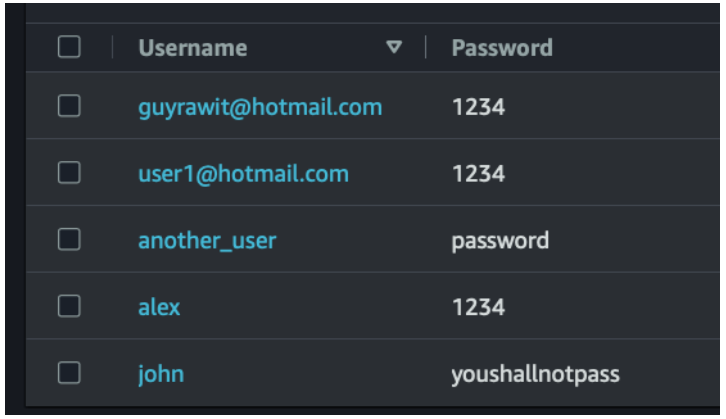
Tables (2) Info

Find tables by table name

Name ▲	Status	Partition key	Sort key	In
myDropboxSharedFiles	Active	shared_user (S)	file_id (S)	
myDropboxUser	Active	Username (S)	-	

Table 1

myDropboxUser
- Partition Key : Username
- Attribute : Password

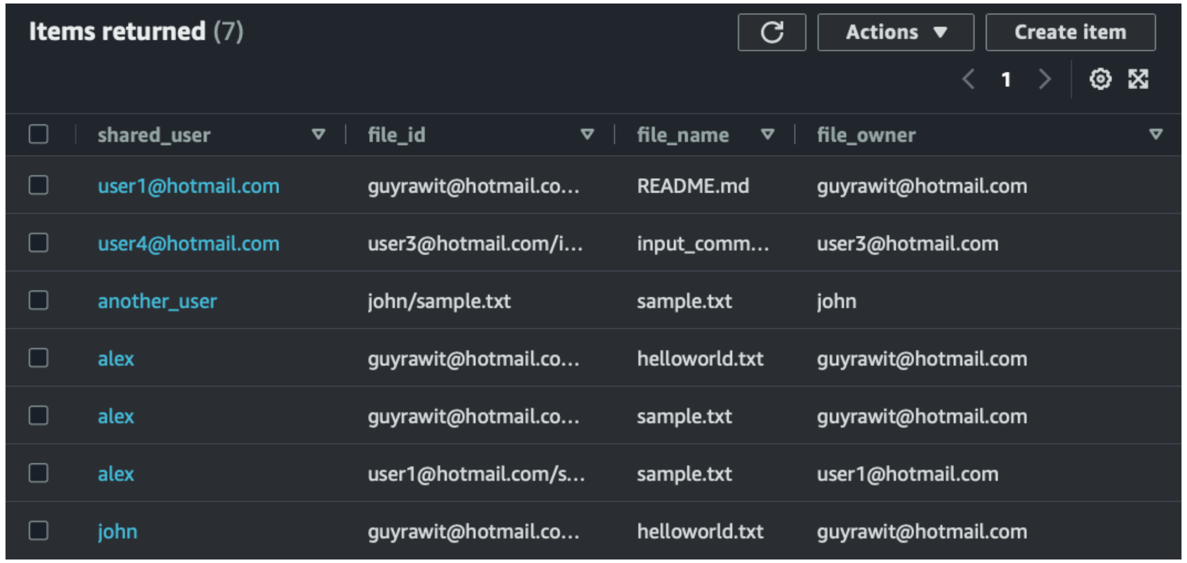


Username	Password
guyrawit@hotmail.com	1234
user1@hotmail.com	1234
another_user	password
alex	1234
john	youshallnotpass

Table 2

Fast and cheap to query files were shared with user

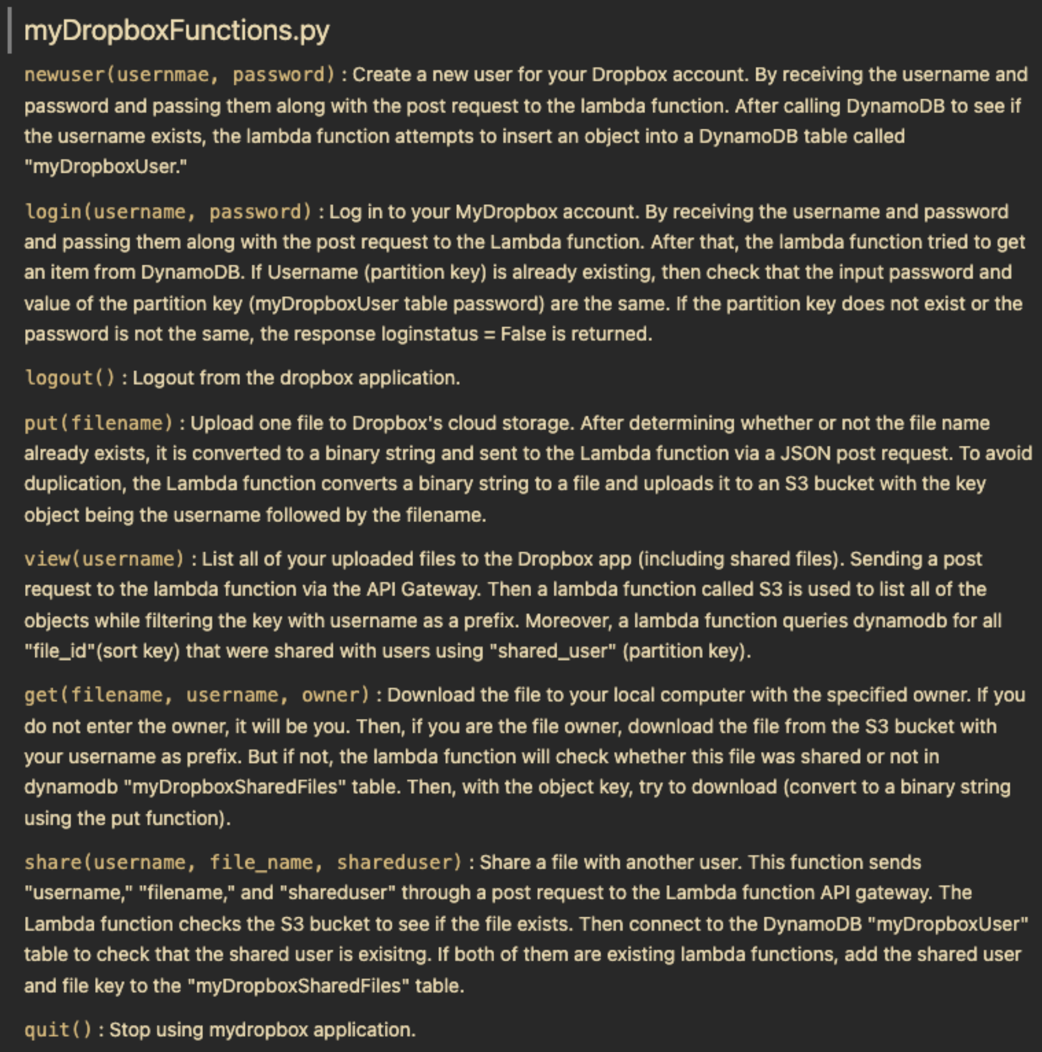
myDropboxSharedFiles
- Partition Key : Shared_user
- Sort Key : File_id
- Attribute : File_name, File_owner



Items returned (7)

shared_user	file_id	file_name	file_owner
user1@hotmail.com	guyrawit@hotmail.co...	README.md	guyrawit@hotmail.com
user4@hotmail.com	user3@hotmail.com/i...	input_comm...	user3@hotmail.com
another_user	john/sample.txt	sample.txt	john
alex	guyrawit@hotmail.co...	helloworld.txt	guyrawit@hotmail.com
alex	guyrawit@hotmail.co...	sample.txt	guyrawit@hotmail.com
alex	user1@hotmail.com/s...	sample.txt	user1@hotmail.com
john	guyrawit@hotmail.co...	helloworld.txt	guyrawit@hotmail.com

MYDROPBOXFUNCTIONS.PY



myDropboxFunctions.py

`newuser(usernmae, password)` : Create a new user for your Dropbox account. By receiving the username and password and passing them along with the post request to the lambda function. After calling DynamoDB to see if the username exists, the lambda function attempts to insert an object into a DynamoDB table called "myDropboxUser."

`login(username, password)` : Log in to your MyDropbox account. By receiving the username and password and passing them along with the post request to the Lambda function. After that, the lambda function tried to get an item from DynamoDB. If Username (partition key) is already existing, then check that the input password and value of the partition key (myDropboxUser table password) are the same. If the partition key does not exist or the password is not the same, the response loginstatus = False is returned.

`logout()` : Logout from the dropbox application.

`put(filename)` : Upload one file to Dropbox's cloud storage. After determining whether or not the file name already exists, it is converted to a binary string and sent to the Lambda function via a JSON post request. To avoid duplication, the Lambda function converts a binary string to a file and uploads it to an S3 bucket with the key object being the username followed by the filename.

`view(username)` : List all of your uploaded files to the Dropbox app (including shared files). Sending a post request to the lambda function via the API Gateway. Then a lambda function called S3 is used to list all of the objects while filtering the key with username as a prefix. Moreover, a lambda function queries dynamodb for all "file_id"(sort key) that were shared with users using "shared_user" (partition key).

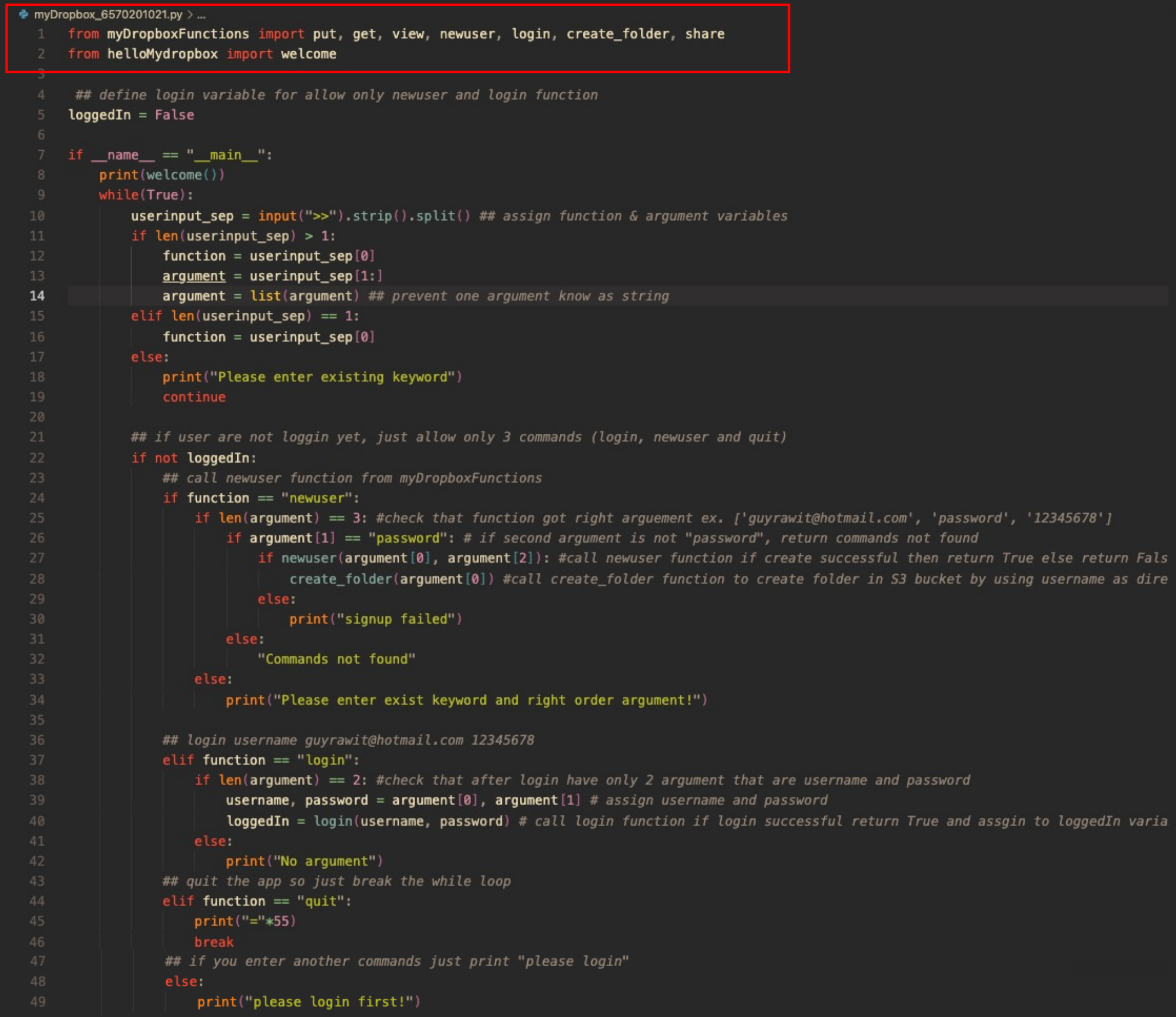
`get(filename, username, owner)` : Download the file to your local computer with the specified owner. If you do not enter the owner, it will be you. Then, if you are the file owner, download the file from the S3 bucket with your username as prefix. But if not, the lambda function will check whether this file was shared or not in dynamodb "myDropboxSharedFiles" table. Then, with the object key, try to download (convert to a binary string using the put function).

`share(username, file_name, shareduser)` : Share a file with another user. This function sends "username," "filename," and "shareduser" through a post request to the Lambda function API gateway. The Lambda function checks the S3 bucket to see if the file exists. Then connect to the DynamoDB "myDropboxUser" table to check that the shared user is exisitng. If both of them are existing lambda functions, add the shared user and file key to the "myDropboxSharedFiles" table.

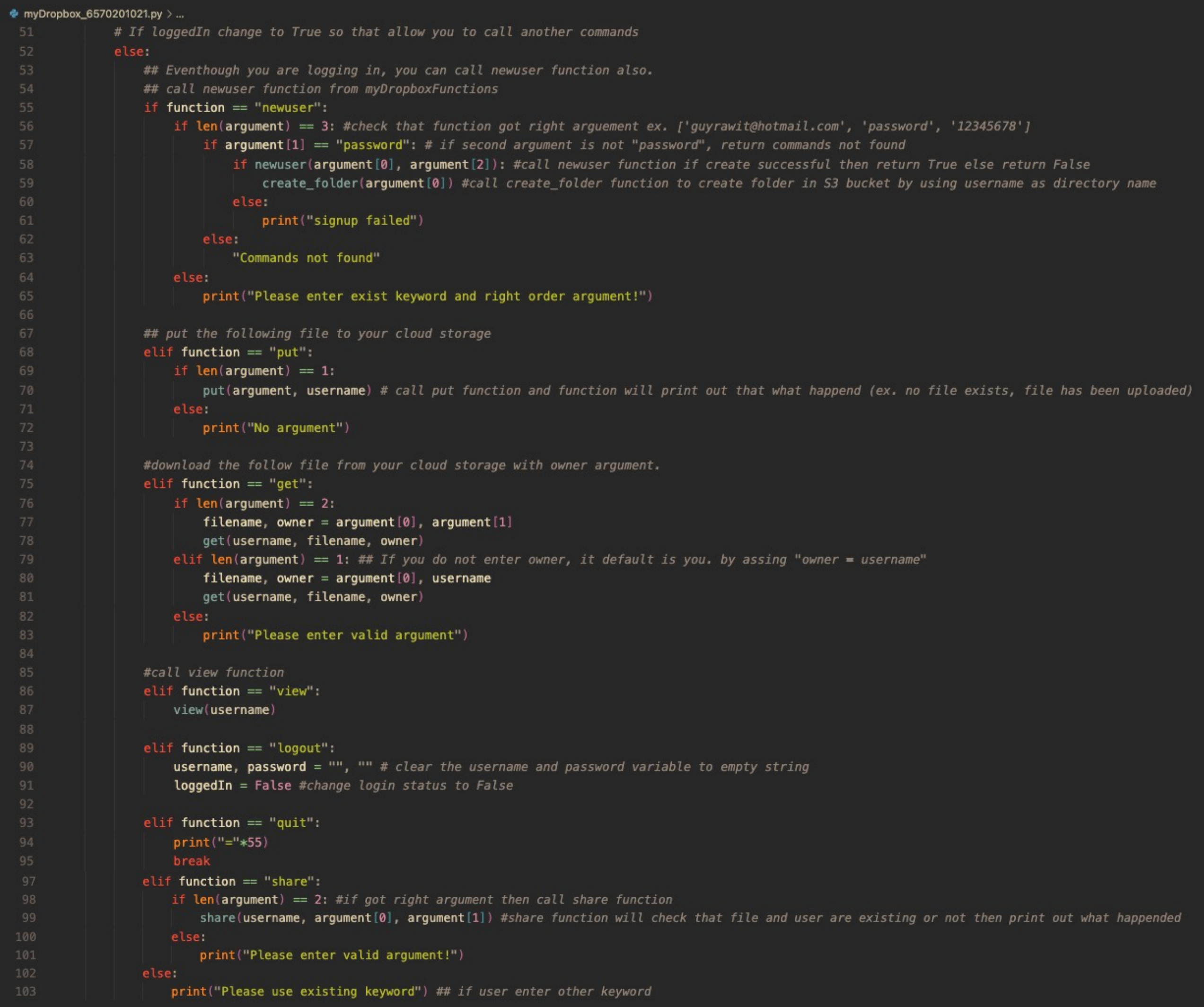
`quit()` : Stop using mydropbox application.

MYDROPBOX_6570201021.PY

Import functions from another python file for more readable



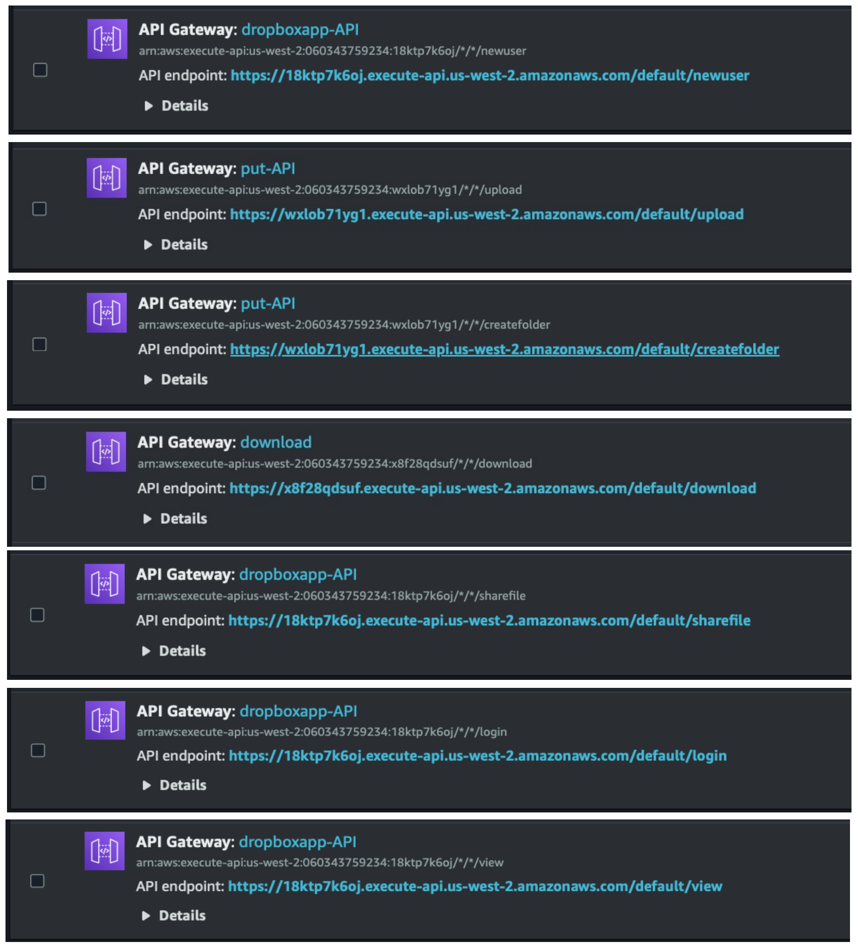
```python
from myDropboxFunctions import put, get, view, newuser, login, create_folder, share
from helloMydropbox import welcome

 ## define login variable for allow only newuser and login function
loggedIn = False

if __name__ == "__main__":
    print(welcome())
    while(True):
        userinput_sep = input(">>").strip().split() ## assign function & argument variables
        if len(userinput_sep) > 1:
            function = userinput_sep[0]
            argument = userinput_sep[1:]
            argument = list(argument) ## prevent one argument know as string
        elif len(userinput_sep) == 1:
            function = userinput_sep[0]
        else:
            print("Please enter existing keyword")
            continue

        ## if user are not loggin yet, just allow only 3 commands (login, newuser and quit)
        if not loggedIn:
            ## call newuser function from myDropboxFunctions
            if function == "newuser":
                if len(argument) == 3: #check that function got right arguement ex. ['guyrawit@hotmail.com', 'password', '12345678']
                    if argument[1] == "password": # if second argument is not "password", return commands not found
                        if newuser(argument[0], argument[2]): #call newuser function if create successful then return True else return Fals
                            create_folder(argument[0]) #call create_folder function to create folder in S3 bucket by using username as dire
                        else:
                            print("signup failed")
                    else:
                        "Commands not found"
                else:
                    print("Please enter exist keyword and right order argument!")

            ## login username guyrawit@hotmail.com 12345678
            elif function == "login":
                if len(argument) == 2: #check that after login have only 2 argument that are username and password
                    username, password = argument[0], argument[1] # assign username and password
                    loggedIn = login(username, password) # call login function if login successful return True and assgin to loggedIn varia
                else:
                    print("No argument")
            ## quit the app so just break the while loop
            elif function == "quit":
                print("="*55)
                break
            ## if you enter another commands just print "please login"
            else:
                print("please login first!")
```



```python
        # If loggedIn change to True so that allow you to call another commands
        else:
            ## Eventhough you are logging in, you can call newuser function also.
            ## call newuser function from myDropboxFunctions
            if function == "newuser":
                if len(argument) == 3: #check that function got right arguement ex. ['guyrawit@hotmail.com', 'password', '12345678']
                    if argument[1] == "password": # if second argument is not "password", return commands not found
                        if newuser(argument[0], argument[2]): #call newuser function if create successful then return True else return False
                            create_folder(argument[0]) #call create_folder function to create folder in S3 bucket by using username as directory name
                        else:
                            print("signup failed")
                    else:
                        "Commands not found"
                else:
                    print("Please enter exist keyword and right order argument!")

            ## put the following file to your cloud storage
            elif function == "put":
                if len(argument) == 1:
                    put(argument, username) # call put function and function will print out that what happend (ex. no file exists, file has been uploaded)
                else:
                    print("No argument")

            #download the follow file from your cloud storage with owner argument.
            elif function == "get":
                if len(argument) == 2:
                    filename, owner = argument[0], argument[1]
                    get(username, filename, owner)
                elif len(argument) == 1: ## If you do not enter owner, it default is you. by assing "owner = username"
                    filename, owner = argument[0], username
                    get(username, filename, owner)
                else:
                    print("Please enter valid argument")

            #call view function
            elif function == "view":
                view(username)

            elif function == "logout":
                username, password = "", "" # clear the username and password variable to empty string
                loggedIn = False #change login status to False

            elif function == "quit":
                print("="*55)
                break
            elif function == "share":
                if len(argument) == 2: #if got right argument then call share function
                    share(username, argument[0], argument[1]) #share function will check that file and user are existing or not then print out what happended
                else:
                    print("Please enter valid argument!")
            else:
                print("Please use existing keyword") ## if user enter other keyword
```

HOWTO

My API Gateway endpoint



API Gateway: dropboxapp-API
arn:aws:execute-api:us-west-2:060343759234:18ktp7k6oj/*/*/newuser
API endpoint: https://18ktp7k6oj.execute-api.us-west-2.amazonaws.com/default/newuser
▶ Details

API Gateway: put-API
arn:aws:execute-api:us-west-2:060343759234:wxlob71yg1/*/*/upload
API endpoint: https://wxlob71yg1.execute-api.us-west-2.amazonaws.com/default/upload
▶ Details

API Gateway: put-API
arn:aws:execute-api:us-west-2:060343759234:wxlob71yg1/*/*/createfolder
API endpoint: https://wxlob71yg1.execute-api.us-west-2.amazonaws.com/default/createfolder
▶ Details

API Gateway: download
arn:aws:execute-api:us-west-2:060343759234:x8f28qdsuf/*/*/download
API endpoint: https://x8f28qdsuf.execute-api.us-west-2.amazonaws.com/default/download
▶ Details

API Gateway: dropboxapp-API
arn:aws:execute-api:us-west-2:060343759234:18ktp7k6oj/*/*/sharefile
API endpoint: https://18ktp7k6oj.execute-api.us-west-2.amazonaws.com/default/sharefile
▶ Details

API Gateway: dropboxapp-API
arn:aws:execute-api:us-west-2:060343759234:18ktp7k6oj/*/*/login
API endpoint: https://18ktp7k6oj.execute-api.us-west-2.amazonaws.com/default/login
▶ Details

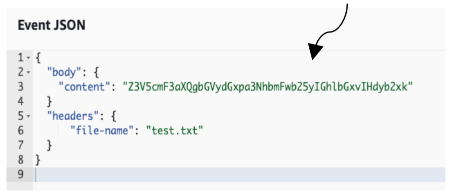
API Gateway: dropboxapp-API
arn:aws:execute-api:us-west-2:060343759234:18ktp7k6oj/*/*/view
API endpoint: https://18ktp7k6oj.execute-api.us-west-2.amazonaws.com/default/view
▶ Details

My Lambda function API format

Put

Binary string

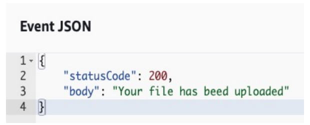
Request format



Event JSON

```
{
  "body": {
    "content": "Z3V5cmF3aXQgbGVydGxpa3NhbmFwb25yIGhlbGxvIHdyb2xk"
  }
  "headers": {
      "file-name": "test.txt"
  }
}
```

Response format

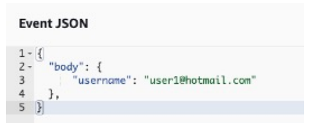


Event JSON

```
{
    "statusCode": 200,
    "body": "Your file has beed uploaded"
}
```

View

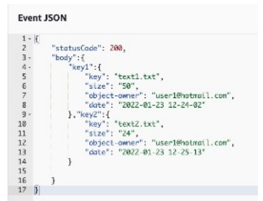
Request format



Event JSON

```
{
  "body": {
     "username": "user1@hotmail.com"
  },
}
```

Response format

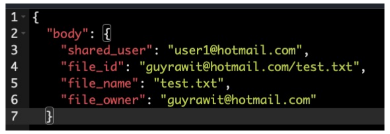


Event JSON

```
{
    "statusCode": 200,
    "body":{
        "key1":{
            "key": "text1.txt",
            "size": "50",
            "object-owner": "user1@hotmail.com",
            "date": "2022-01-23 12-24-02"
        },"key2":{
            "key": "text2.txt",
            "size": "24",
            "object-owner": "user1@hotmail.com",
            "date": "2022-01-23 12-25-13"
        }

    }
}
```

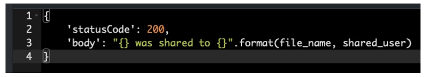
Shared file

Request format



```
{
  "body": {
    "shared_user": "user1@hotmail.com",
    "file_id": "guyrawit@hotmail.com/test.txt",
    "file_name": "test.txt",
    "file_owner": "guyrawit@hotmail.com"
  }
```

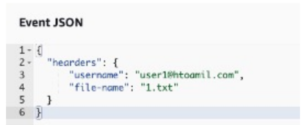
Response format



```
{
    'statusCode': 200,
    'body': "{} was shared to {}".format(file_name, shared_user)
}
```

Get

Request format

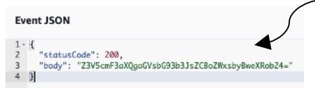


Event JSON

```
{
  "hearders": {
      "username": "user1@htoamil.com",
      "file-name": "1.txt"
  }
}
```

Binary string

Response format

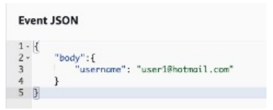


Event JSON

```
{
  "statusCode": 200,
  "body": "Z3V5cmF3aXQgaGVsbG93b3JsZCBoZWxsbyBweXRob24="
}
```

Create_folder

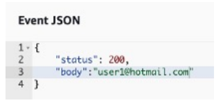
Request format



Event JSON

```
{
    "body":{
        "username": "user1@hotmail.com"
    }
}
```

Response format

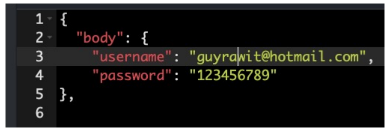


Event JSON

```
{
    "status": 200,
    "body":"user1@hotmail.com"
}
```

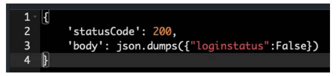
Login

Request format



```
{
  "body": {
    "username": "guyrawit@hotmail.com",
    "password": "123456789"
},

```

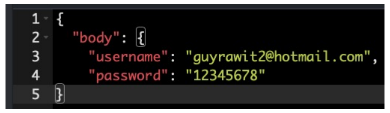
Response format



```
{
    'statusCode': 200,
    'body': json.dumps({"loginstatus":False})
}
```

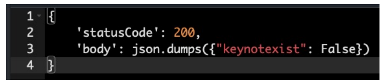
Newuser

Request format



```
{
  "body": {
    "username": "guyrawit2@hotmail.com",
    "password": "12345678"
}
```

Response format



```
{
    'statusCode': 200,
    'body': json.dumps({"keynotexist": False})
}
```