Python Assignment - TAS-263 Chandra Sai D

Implement s3 file manager using any python web framework(flask/django/...etc).
functions :
1. List content of s3.
2. Create/Delete folder + bucket .
3. Upload files to s3 + delete file from s3.
4. Copy/Move file withing s3.
Note:
1. Make sure your code is readable
2. Make sure your app is working properly
3. Need basic UI from which we can access app

Steps to create a AWS S3 bucket:

1. Visit AWS website and create an AWS account and provide all the details for the creation of your account.
2. We have create an user in IAM (Identity and Access management) configuration.
3. Provide the necessary details for the creation of user and generate the access key and download the .csv file of credentials for further use.
4. We have to create a S3 bucket and provide the necessary details for the bucket creation.

Steps to create the project:

1. Download any python IDE (VSCode/PyCharm) in the terminal and create a new project file.
2. We need to install the flask and boto3 package
   -> pip install flask boto3 flask-wtf
3. Create the app.py file for the flask to write and run the operations mentioned like creating the folder, uploading the files, listing the files, move and copy the files from source to destination.

   Code:

   #Importing the necessary packages from flask

```
from flask import Flask, render_template, request, redirect, url_for,
flash
import boto3

app = Flask(__name__)
app.secret_key = 'your_secret_key'
```

```
AWS_ACCESS_KEY = 'AKIA6JKEYGAG7DCEHZLV'
AWS_SECRET_ACCESS_KEY = 'V2ZxgiuCf2H9sjhkproouPJVTpTpTQltH3zYLM5P'
REGION = 'eu-north-1'
S3_BUCKET = 'chandrasai'
# Initialize S3 client
s3 = boto3.client(
   "s3",
   aws_access_key_id=AWS_ACCESS_KEY,
   aws_secret_access_key=AWS_SECRET_ACCESS_KEY,
   region_name=REGION
)
```

# Route to list contents of the S3 bucket

```
@app.route('/')
def index():
   contents = list_s3_content()
   folders = get_folders(contents)
   return render_template('index.html', contents=contents,
folders=folders)
```

# List all objects and folders in the bucket

```
def list_s3_content():
   try:
       response = s3.list_objects_v2(Bucket=S3_BUCKET)
       contents = response.get('Contents', [])
       return contents
   except Exception as e:
       flash(f"Error: {str(e)}")
       return []
```

# Extract folder names from S3 contents

```
def get_folders(contents):
   folders = set()
   for item in contents:
       key = item['Key']
       if key.endswith('/'):
           folders.add(key)
   return sorted(folders)
```

# Route to create a folder

```
@app.route('/create-folder', methods=['POST'])
def create_folder():
   folder_name = request.form['folder_name']
   if folder_name:
       folder_name = folder_name.rstrip('/') + '/'
       try:
           s3.put_object(Bucket=S3_BUCKET, Key=folder_name)
```

```
            flash('Folder created successfully!')
        except Exception as e:
            flash(f"Error: {str(e)}")
    return redirect(url_for('index'))
```
# Route to delete a folder or file
```
@app.route('/delete', methods=['POST'])
def delete_object():
    key = request.form['key']
    try:
        s3.delete_object(Bucket=S3_BUCKET, Key=key)
        flash('Deleted successfully!')
    except Exception as e:
        flash(f"Error: {str(e)}")
    return redirect(url_for('index'))
```

# Route to upload a file with selected folder
```
@app.route('/upload', methods=['POST'])
def upload_file():
    folder = request.form.get('folder')
    if 'file' not in request.files:
        flash('No file part')
        return redirect(url_for('index'))

    file = request.files['file']
    if file.filename == '':
        flash('No selected file')
        return redirect(url_for('index'))

    if folder:
        file_key = f"{folder}{file.filename}"
    else:
        file_key = file.filename

    try:
        s3.upload_fileobj(file, S3_BUCKET, file_key)
        flash('File uploaded successfully!')
    except Exception as e:
        flash(f"Error: {str(e)}")

    return redirect(url_for('index'))
```

# Route to move or copy a file
```
@app.route('/move-copy', methods=['POST'])
def move_copy_file():
    src_key = request.form['src_key']
    dest_key = request.form['dest_key']
    action = request.form['action']

    try:
        if action == 'copy':
            s3.copy_object(Bucket=S3_BUCKET, CopySource={'Bucket':
S3_BUCKET, 'Key': src_key}, Key=dest_key)
            flash('File copied successfully!')
        elif action == 'move':
            s3.copy_object(Bucket=S3_BUCKET, CopySource={'Bucket':
S3_BUCKET, 'Key': src_key}, Key=dest_key)
            s3.delete_object(Bucket=S3_BUCKET, Key=src_key)
            flash('File moved successfully!')
    except Exception as e:
        flash(f"Error: {str(e)}")

    return redirect(url_for('index'))
```

#To run the app.py

```
if __name__ == "__main__":
    app.run(debug=True,port='5055')
```

4. We have to create an basic html file for the creation of frontend based on the requirements.

Code: index.html

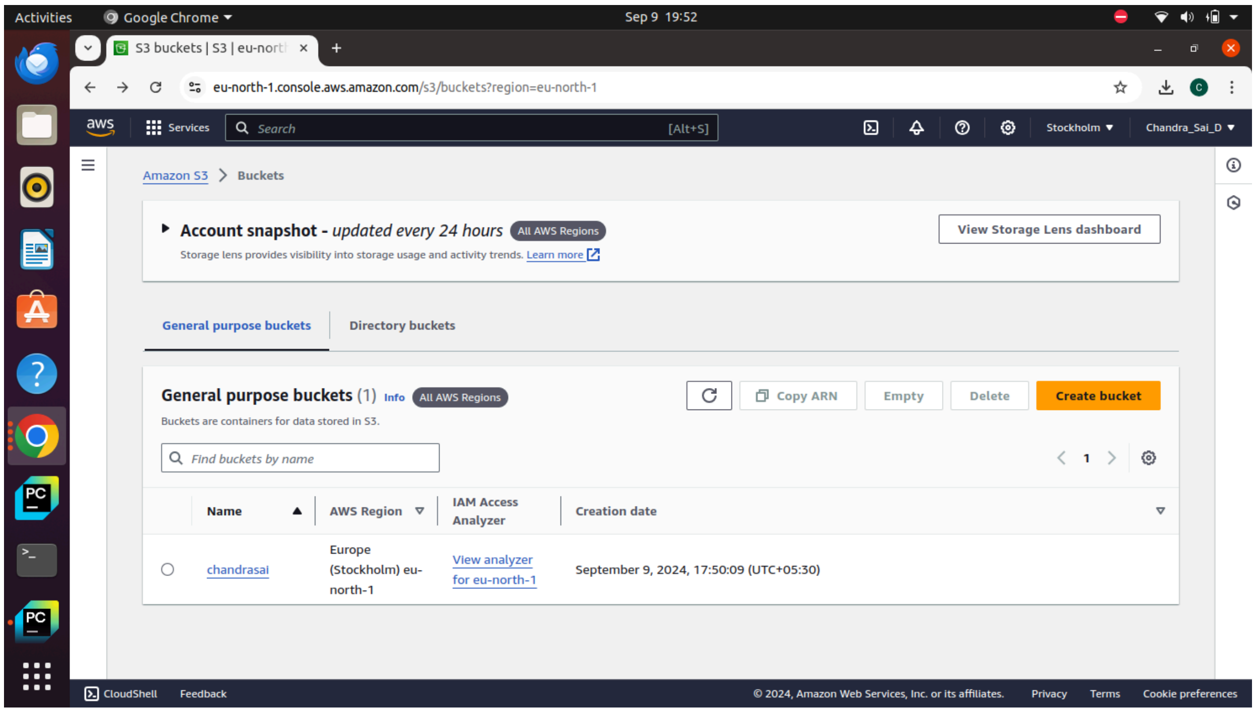
```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>S3 File Manager - Custom Design</title>
 <link rel="preconnect" href="https://fonts.googleapis.com">
 <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
 <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600;700&display
=swap" rel="stylesheet">
 <style>
<body>
 <div class="container">
   <div class="header">
     <h1>S3 File Manager</h1>
     <p>Manage your files and folders in Amazon S3 with ease.</p>
   </div>

   <!-- Flash messages -->
   {% with messages = get_flashed_messages() %}
     {% if messages %}
       <div class="alert alert-info">
         {{ messages[0] }}
       </div>
     {% endif %}
   {% endwith %}
```

```
<!-- List Files and Folders -->
<div class="card">
  <div class="card-header">
    Files and Folders
  </div>
  <div class="card-body">
    <ul class="file-list">
      {% for item in contents %}
        <li>
          <span>{{ item.Key }}</span>
          <form action="/delete" method="POST" class="mb-0">
            <input type="hidden" name="key" value="{{ item.Key }}">
            <button type="submit" class="btn">Delete</button>
          </form>
        </li>
      {% endfor %}
    </ul>
  </div>
</div>
```

Contd.

Snapshots of the result:



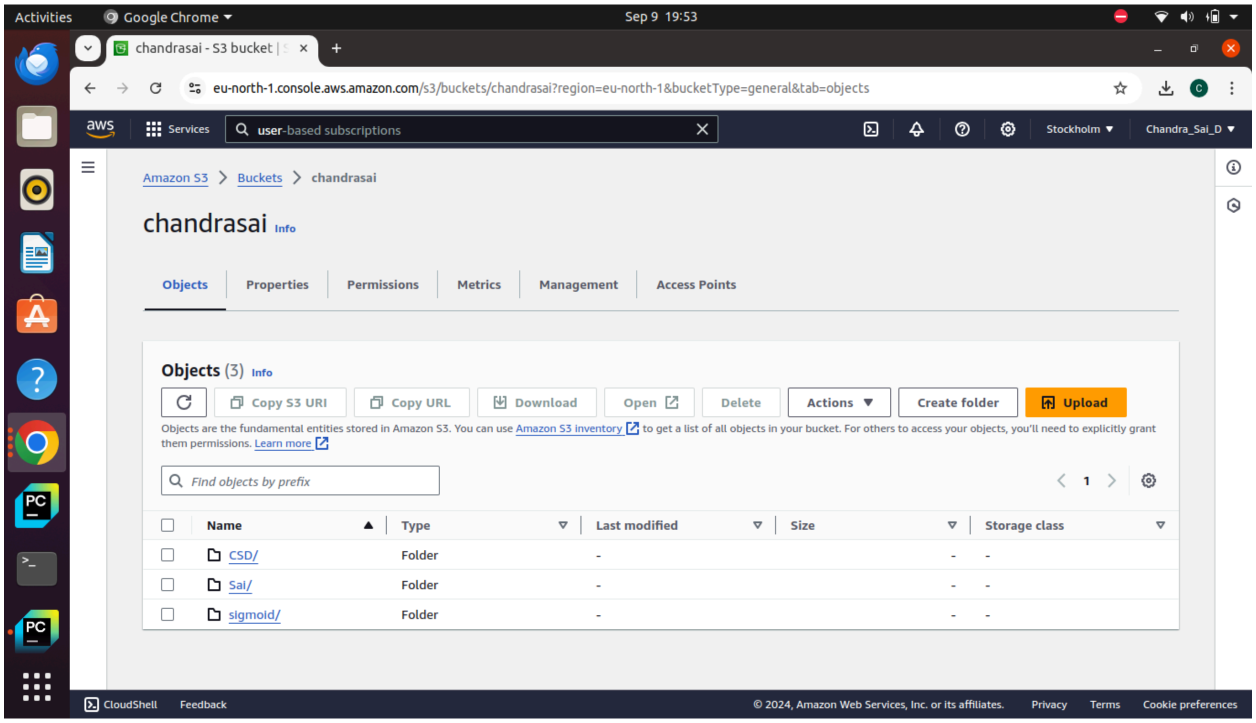
Storage lens provides visibility into storage usage and activity trends. Learn more

General purpose buckets (1) All AWS Regions

Buckets are containers for data stored in S3.

Name	AWS Region	IAM Access Analyzer	Creation date
chandrasai	Europe (Stockholm) eu-north-1	View analyzer for eu-north-1	September 9, 2024, 17:50:09 (UTC+05:30)

© 2024, Amazon Web Services, Inc. or its affiliates.


Amazon S3 > Buckets > chandrasai

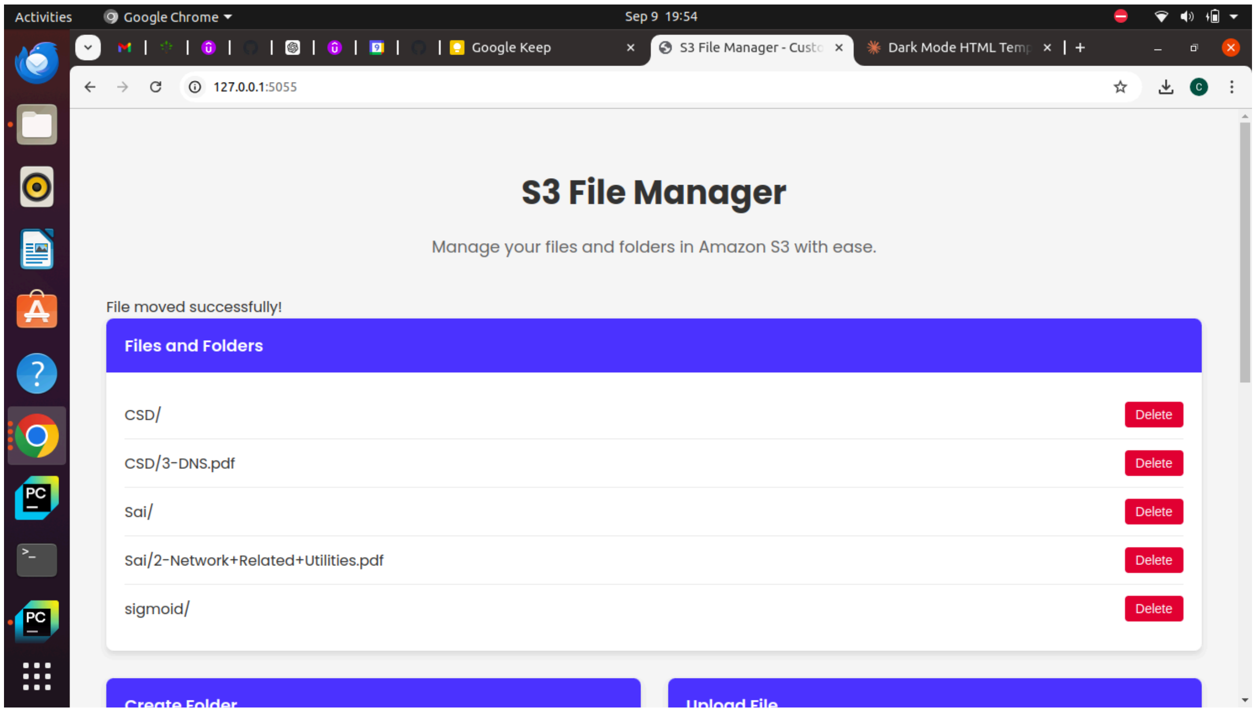
chandrasai Info

Objects | Properties | Permissions | Metrics | Management | Access Points

Objects (3) Info

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
CSD/	Folder	-	-	-
Sai/	Folder	-	-	-
sigmoid/	Folder	-	-	-


S3 File Manager

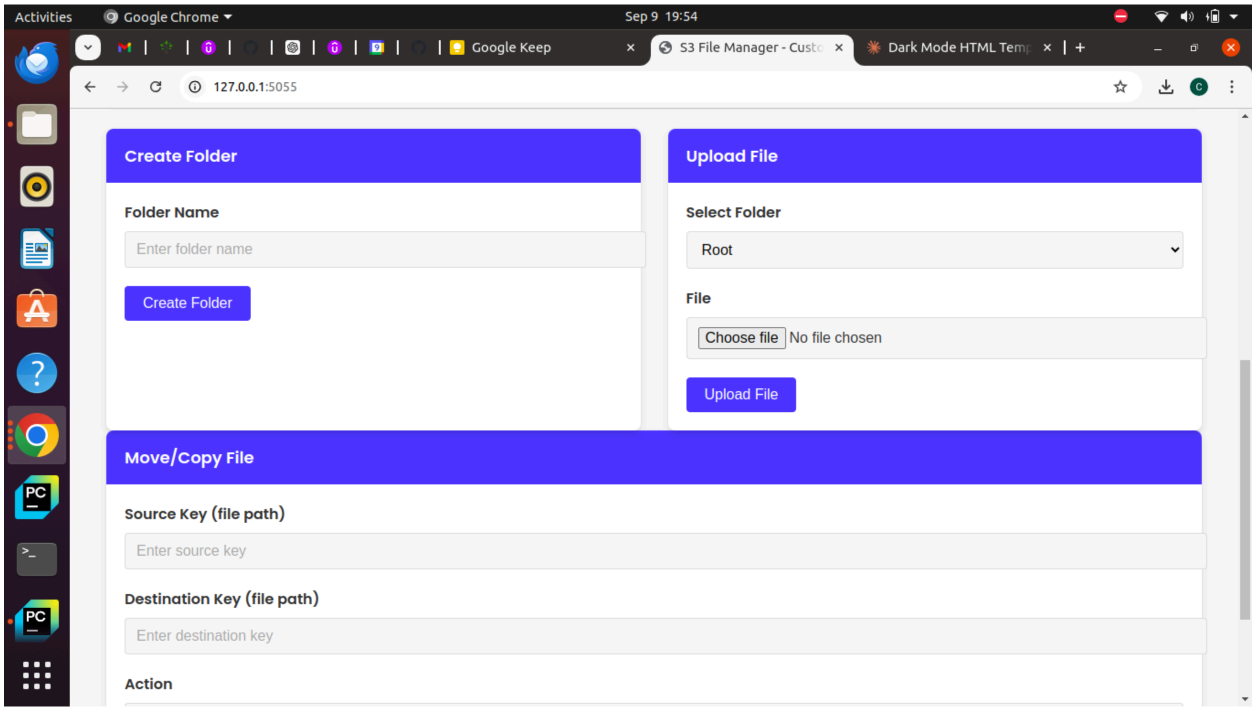
Manage your files and folders in Amazon S3 with ease.

File moved successfully!

Files and Folders

```
CSD/                                  Delete
CSD/3-DNS.pdf                         Delete
Sai/                                  Delete
Sai/2-Network+Related+Utilities.pdf   Delete
sigmoid/                              Delete
```

Create Folder   Upload File



Create Folder

Folder Name

Enter folder name

Create Folder

Upload File

Select Folder

Root

File

Choose file No file chosen

Upload File

Move/Copy File

Source Key (file path)

Enter source key

Destination Key (file path)

Enter destination key

Action