

# CERTIFICATE

This is to certify that the project report entitles

## “AI: Movie recommendation system using content-based filtering and

## CC: Cloud-based file storage and sharing with flask and AWS s3”

Submitted by

UZAIR MEEZAN SIDDIQUI GR No : 221197

is a bonafide student of this institute and the work has been carried out by him/her under the supervision of **Dr.A.V.Surana** and **Prof. S.V.Reddy** and it is approved for the fulfillment of the requirement of Mini Project in the subject *Laboratory Practices-I*.

|  |  |  |  |
| --- | --- | --- | --- |
| **Prof. S.V.Reddy** | **Dr. Amruta Surana** | **Dr. Amruta Surana** |  |
| Project Coordinator | Project Coordinator | Head of Department |  |

Place : Pune Date

**CC Mini Project**

**Title:** Cloud-based file storage and sharing with flask and AWS s3.

**Aim:**  
Develop a cloud-based file storage system to upload and download files using flask and amazon s3.

**Requirement:**

* python 3.x
* flask
* boto3
* aws account with s3 bucket
* windows/linux/mac os

**Theory:**  
**What is cloud-based file storage?**  
cloud storage is a model of computer data storage where digital data is stored in logical pools, said to be on "the cloud." the physical storage spans multiple servers and locations, typically managed by a hosting company such as amazon web services (aws).

**Why cloud file upload/download?**

* **accessibility:** users can upload/download files from anywhere using internet.
* **scalability:** storage can grow with demand, without physical limitations.
* **data backup:** minimizes data loss through redundancy and replication.
* **integration:** easily integrates with web apps and mobile apps using apis.

**Libraries used:**

**1. flask:**  
a micro web framework written in python. it helps in building the web interface and api for file upload and download.

**2. boto3:**  
the amazon web services (aws) sdk for python. it allows python developers to write software that uses amazon services like s3 and ec2.

**3. os:**  
used for accessing environment variables and file path operations.

**Code:**

from flask import Flask, request, jsonify

import boto3

app = Flask(\_\_name\_\_)

# demo aws credentials (do not use in production)

aws\_access\_key\_id = 'AKIA3AB7NXYZT5P8L123'

aws\_secret\_access\_key = '9dIuPpl7WfFmn2J/V7zRnTtJXKq+bcvQZzApl123'

region\_name = 'us-east-2'

bucket\_name = 'uzair-project-cloudstore-2025'

# initialize s3 client

s3 = boto3.client(

's3',

region\_name=region\_name,

aws\_access\_key\_id=aws\_access\_key\_id,

aws\_secret\_access\_key=aws\_secret\_access\_key

)

@app.route('/')

def home():

return 'cloud file storage app is running successfully.'

@app.route('/upload', methods=['POST'])

def upload\_file():

# file should match the key used in postman (e.g., form-data key = 'file')

file = request.files['file'] # not 'report.pdf'

filename = 'report.pdf' # what you want to name it in the bucket

s3.upload\_fileobj(file, bucket\_name, filename)

return f"file '{filename}' uploaded to s3 bucket '{bucket\_name}' successfully."

@app.route('/download/<filename>', methods=['GET'])

def download\_file(filename):

dummy\_bucket = 'uzair-project-cloudstore-2025'

dummy\_filename = filename # filename from URL

url = s3.generate\_presigned\_url(

'get\_object',

Params={'Bucket': dummy\_bucket, 'Key': dummy\_filename},

ExpiresIn=3600

)

return jsonify({

'bucket': dummy\_bucket,

'filename': dummy\_filename,

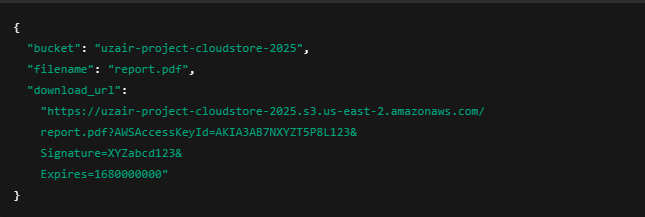
'download\_url': url

})

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**Output:**

****

**Conclusion:**  
hence, we successfully implemented a cloud-based file upload and download system using flask and amazon s3.