PROFESSIONAL TRAINING REPORT
at

Sathyabama Institute of Science and Technology (Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering

By

LEVAKU VENKATA KOWSHIK REDDY

(Reg No : 40731054)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU

OCTOBER 2022

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI– 600119

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **LEVAKU VENKATA KOWSHIK REDDY (Reg No : 40731054)** who carried out the project entitled"**AWS Storage**" under my supervision from Aug 2022 to Oct 2022.

Internal Guide

Dr. J.Albert Mayan M.E., Ph.D.,

Head of the Department

Dr. L.Lakshmanan M.E., Ph.D.,

Submitted for Viva voce Examination held on____________________

Internal Examiner **External Examiner**

DECLARATION

I,**LEVAKU VENKATA KOWSHIK REDDY** hereby declare that the Project Report entitled **"AWS Storage"** done by me under the guidance of **Dr. J. Albert Mayan** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: **SIGNATURE OF THECANDIDATE**

ACKNOWLEDGEMENT

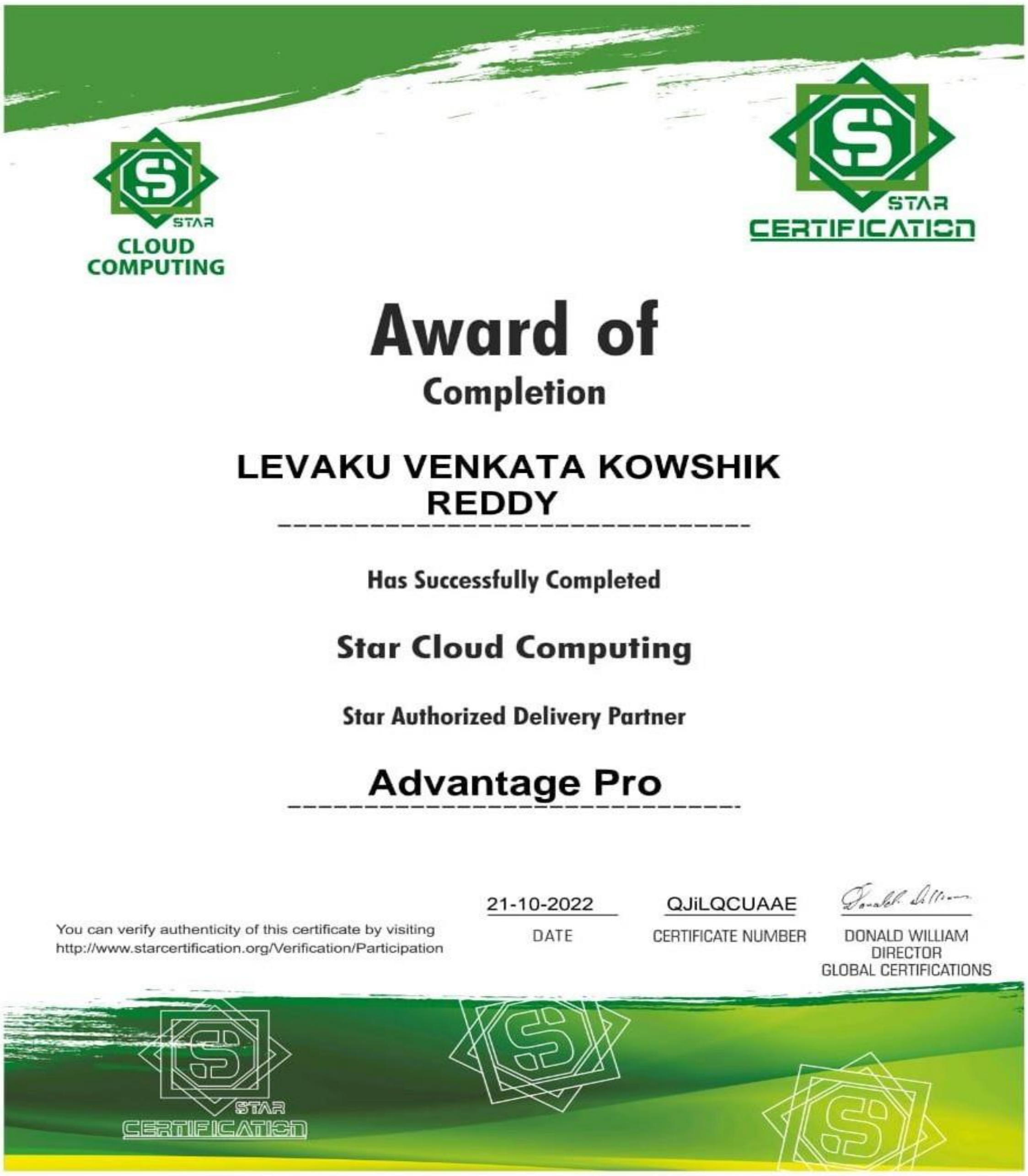
I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr.T.Sasikala M.E.,Ph.D**, **Dean**, School of Computing, **Dr.L.Lakshmanan M.E., Ph.D.,** Head of the Department of Computer Science and Engineering for providing me necessary supportand details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.J.Albert Mayan** ,for his valuable guidance, suggestions and constant encouragementpaved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of Computer Science and Engineering who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE



STAR
CLOUD
COMPUTING

STAR
CERTIFICATION

Award of
Completion

LEVAKU VENKATA KOWSHIK
REDDY

Has Successfully Completed

Star Cloud Computing

Star Authorized Delivery Partner

Advantage Pro

You can verify authenticity of this certificate by visiting
http://www.starcertification.org/Verification/Participation

21-10-2022	QJiLQCUAAE	[illegible]
DATE	CERTIFICATE NUMBER	DONALD WILLIAM DIRECTOR GLOBAL CERTIFICATIONS

STAR
CERTIFICATION

ABSTRACT

Cloud Computing is going to be the future and way to store each and every data's in the present world. It Conveniently opens your application in different regions around the world with just a few clicks. As a result, the customer can experience lower latency and low cost.Cloud Computing is the finest way to store and access all user data. When it comes to Cloud Computing "Amazon Web Services (AWS)" is the popular cloud services platform.AWS is used by many organizations and individuals all over the world.AWS is everywhere in the world and it has been providing more than 100+ services in various types.AWS supports a highly safe, extensible, low-cost infrastructure platform in the cloud.AWS having a variety of big data analytics and application services.AWS has the possibility of being the age defining a cloud service provider.It further brings attention and expectation of the cloud consumers/service providers. People are willing to Cooperate with AWS due to the center of attraction built around it

TABLE OF CONTENTS

CHAPTER No	TITLE	PAGE No
	ABSTRACT	
	LIST OF FIGURES	
1	**INTRODUCTION**	
	1.1 AWS STORAGE	1
	1.2 AMAZON S3	1
	1.3 BUCKET IN S3	1
	1.4 FEATURES OF S3 BUCKET	2
	1.4.1 STORAGE CLASSES	2
	1.4.2 BUCKET POLICY	5
2.	**AIM AND SCOPE OF THE PRESENT INVESTIGATION**	
	2.1 AIM	6
	2.2 SCOPE	6
	2.3 HOW AMAZON S3 WORKS	6

3	**EXPERIMENTAL OR MATERIALS AND METHODS, ALGORITHMS USED**	
	3.1 MATERIAL USED	7
	3.2 IMPORTED LIBRARIES	8
	3.3 BUCKET CREATION	9
	3.4 VERSIONING	9
	3.5 ENCRYPTION	10
	3.6 ACL	11
	3.7 BUCKET POLICY	11
4	**RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS**	
	4.1 MODEL ANALYSIS	12
5	**SUMMARY AND CONCLUSIONS**	
	5.1 SUMMARY AND CONCLUSIONS	14
	REFERENCES	15
	APPENDIX SCREENSHOTS AND OUTPUTS	17

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO
1.1	types of storage classes	2
3.1.1	installation of awscli and aws configure	7
3.2.1	boto3	8
3.2.2	importing libraries	9
3.3	creating bucket	9
3.4	enabling bucket versioning	9
3.4.1	how versioning works	10
3.5	bucket encryption	10
3.6	acl	11
3.7	bucket policy	11
4.1.1	diff types of storage classes	12
4.1.2	pie chart for usage of different cloud services	13
5.1	summary of aws	14

CHAPTER 1
INTRODUCTION

1.1 AWS STORAGE

AWS offers a wide range of storage services that can be provisioned depending on your project requirements and use case. AWS storage services have different provisions for highly confidential data, frequently accessed data, and the not so frequently accessed data. You can choose from various storage types namely, object storage, file storage, block storage services, backups, and *data* migration options. All of which fall under the AWS Storage Services list.

1.2 AMAZON S3

S3, is the object storage service provided by AWS. It is probably the most commonly used, go-to storage service for AWS users given the features like extremely high availability, security, and simple connection to other AWS Services. AWS S3 can be used by people with all kinds of use cases like mobile/web applications, big data, machine learning and many more.

1.3 BUCKET IN S3

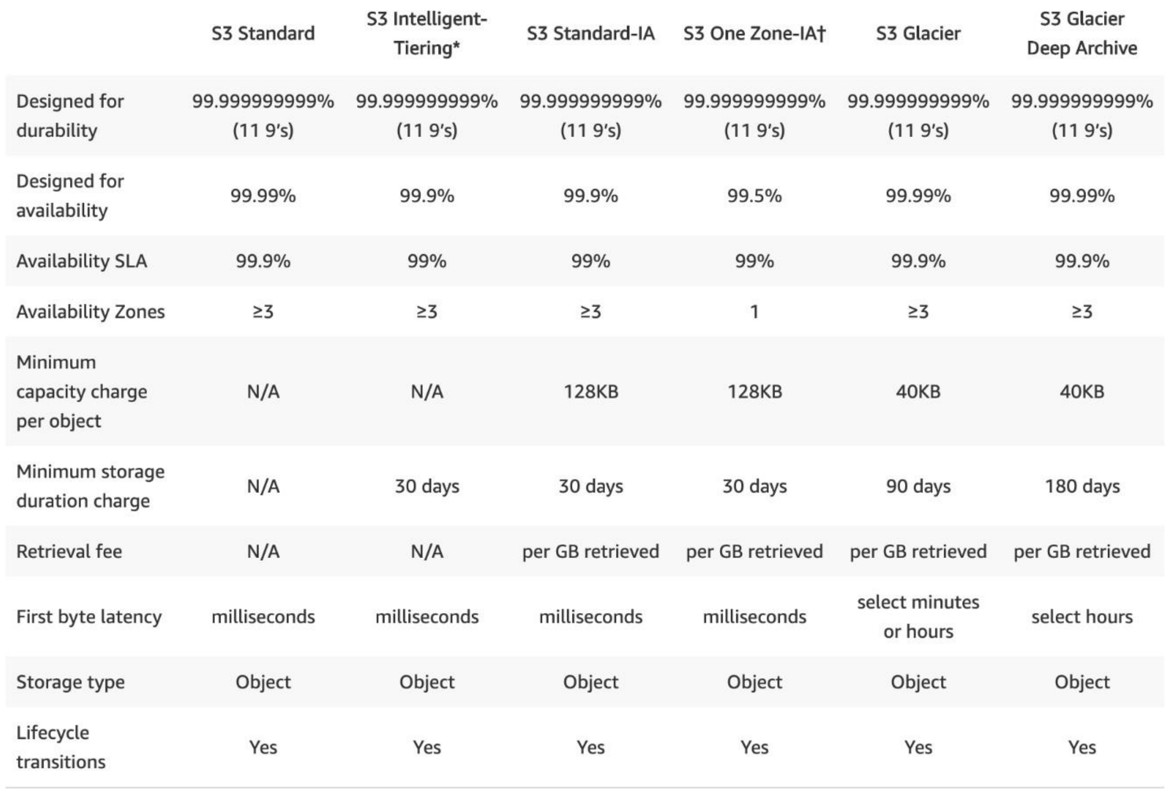
In S3 data is stored in containers called buckets.

- Each bucket will have its own set of policies and configuration.
- This enables users to have more control over their data.
- Bucket Names must be unique.

1.4 FEATURES OF S3 BUCKET

1.4.1 STORAGE CLASSES:

Amazon S3 offers a range of storage classes designed for different use cases. For example, you can store mission-critical production data in S3 Standard for frequent access, save costs by storing infrequently accessed data in S3 Standard-IA or S3 One Zone-IA, and archive data at the lowest costs in S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval, and S3 Glacier Deep Archive. You can store data with changing or unknown access patterns in S3 Intelligent-Tiering, which optimizes storage costs by automatically moving your data between four access tiers when your access patterns change. These four access tiers include two low-latency access tiers optimized for frequent and infrequent access, and two opt-in archive access tiers designed for asynchronous access for rarely accessed data



	S3 Standard	S3 Intelligent-Tiering*	S3 Standard-IA	S3 One Zone-IA†	S3 Glacier	S3 Glacier Deep Archive
Designed for durability	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)
Designed for availability	99.99%	99.9%	99.9%	99.5%	99.99%	99.99%
Availability SLA	99.9%	99%	99%	99%	99.9%	99.9%
Availability Zones	≥3	≥3	≥3	1	≥3	≥3
Minimum capacity charge per object	N/A	N/A	128KB	128KB	40KB	40KB
Minimum storage duration charge	N/A	30 days	30 days	30 days	90 days	180 days
Retrieval fee	N/A	N/A	per GB retrieved	per GB retrieved	per GB retrieved	per GB retrieved
First byte latency	milliseconds	milliseconds	milliseconds	milliseconds	select minutes or hours	select hours
Storage type	Object	Object	Object	Object	Object	Object
Lifecycle transitions	Yes	Yes	Yes	Yes	Yes	Yes

Fig 1.1-types of storage classes

S3 STANDARD CLASS:

S3 Standard offers high durability, availability, and performance object storage for frequently accessed data. Because it delivers low latency and high throughput, S3 Standard is appropriate for a wide variety of use cases, including cloud applications, dynamic websites, content distribution, mobile and gaming applications, and big data analytics.

S3 INTELLIGENT TIERING CLASS:

Amazon S3 Intelligent-Tiering (S3 Intelligent-Tiering) is the first cloud storage that automatically reduces your storage costs on a granular object level by automatically moving data to the most cost-effective access tier based on access frequency, without performance impact, retrieval fees, or operational overhead. S3 Intelligent-Tiering delivers milliseconds latency and high throughput performance for frequently, infrequently, and rarely accessed data in the Frequent, Infrequent, and Archive Instant Access tiers. You can use S3 Intelligent-Tiering as the default storage class for virtually any workload, especially data lakes, data analytics, new applications, and user-generated content.

S3 STANDARD-INFREQUENT ACCESS (S3 STANDARD-IA) CLASS:

S3 Standard-IA is for data that is accessed less frequently, but requires rapid access when needed. S3 Standard-IA offers the high durability, high throughput, and low latency of S3 Standard, with a low per GB storage price and per GB retrieval charge. This combination of low cost and high performance make S3 Standard-IA ideal for long-term storage, backups, and as a data store for disaster recovery files.

S3 ONE ZONE INFREQUENT ACCESS (S3 ONE ZONE--IA) CLASS:

S3 One Zone-IA is for data that is accessed less frequently, but requires rapid access when needed. Unlike other S3 Storage Classes which store data in a minimum of three Availability Zones (AZs), S3 One Zone-IA stores data in a single AZ and costs 20% less than S3 Standard-IA. S3 One Zone-IA is ideal for customers who want a lower-cost option for infrequently accessed data but do not require the availability and resilience of S3 Standard or S3 Standard-IA. It’s a good choice for storing secondary backup copies of on-premises data or easily re-creatable data. You can also use it as cost-effective storage for data that is replicated from another AWS Region using S3 Cross-Region Replication.

S3 GLACIER CLASS:

Amazon S3 Glacier storage classes are purpose-built for data archiving, providing you with the highest performance, most retrieval flexibility, and the lowest cost archive storage in the cloud. All S3 Glacier storage classes provide virtually unlimited scalability and are designed for 99.999999999% (11 nines) of data durability. The S3 Glacier storage classes deliver options for the fastest access to your archive data and the lowest-cost archive storage in the cloud.

S3 GLACIER DEEP ARCHIVE:

S3 Glacier Deep Archive is Amazon S3’s lowest-cost storage class and supports long-term retention and digital preservation for data that may be accessed once or twice in a year. It is designed for customers—particularly those in highly-regulated industries, such as financial services, healthcare, and public sectors—that retain data sets for 7—10 years or longer to meet regulatory compliance requirements. S3 Glacier Deep Archive can also be used for backup and disaster recovery use cases, and is a cost-effective and easy-to-manage alternative to magnetic tape systems, whether they are on-premises libraries or off-premises services.

1.4.2 BUCKET POLICY

A bucket policy is a resource-based policy that you can use to grant access permissions to your bucket and the objects in it. Only the bucket owner can associate a policy with a bucket. The permissions attached to the bucket apply to all of the objects in the bucket that are owned by the bucket owner. These permissions do not apply to objects owned by other AWS accounts.

In its most basic sense, a bucket policy contains the following elements:

- Resources – Buckets, objects, cess points, and jobs are the Amazon S3 resources for which you can allow or deny permissions. In a policy, you use the Amazon Resource Name (ARN) to identify the resource. For more information, see Amazon S3 resources.
- Actions – For each resource, Amazon S3 supports a set of operations. You identify resource operations that you will allow (or deny) by using action keywords.
- Effect – What the effect will be when the user requests the specific action—this can be either *allow* or *deny.*

  If you do not explicitly grant access to (allow) a resource, access is implicitly denied. You can also explicitly deny access to a resource. You might do this to make sure that a user can't access the resource, even if a different policy grants access. For more information, see IAM JSON Policy Elements: Effect.
- Principal – The account or user who is allowed access to the actions and resources in the statement. In a bucket policy, the principal is the user, account, service, or other entity that is the recipient of this permission. For more information, see Principals.
- Condition – Conditions for when a policy is in effect. You can use AWS-wide keys and Amazon S3-specific keys to specify conditions in an Amazon S3 access policy.

CHAPTER 2
AIM AND SCOPE OF PRESENT INVESTIGATION

2.1 AIM:

The aim of Amazon S3 is to store and retrieve any amount of data using highly scalable, reliable, fast and inexpensive data storage.

2.2 SCOPE:

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can use Amazon S3 to store and protect any amount of data for a range of use cases, such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides management features so that you can optimize, organize, and configure access to your data to meet your specific business, organizational, and compliance requirements.Amazon S3 offers features to help you gain visibility into your storage usage, which empowers you to better understand, analyze, and optimize your storage at scale

2.3 HOW MANY AMAZON S3 WORKS:

Amazon S3 is an object storage service that stores data as objects within buckets. An object is a file and any metadata that describes the file. A bucket is a container for objects. To store your data in Amazon S3, you first create a bucket and specify a bucket name and AWS Region. Then, you upload your data to that bucket as objects in Amazon S3. Each object has a key (or key name), which is the unique identifier for the object within the bucket. S3 provides features that you can configure to support your specific use case. For example, you can use S3 Versioning to keep multiple versions of an object in the same bucket, which allows you to restore objects that are accidentally deleted or overwritten. Buckets and the objects in them are private and can be accessed only if you explicitly grant access permissions. You can use bucket policies, AWS Identity and Access Management (IAM) policies, access control lists (ACLs), and S3 Access Points to manage access.

CHAPTER 3
EXPERIMENTAL OR MATERIAL AND METHODS ALGORITHMS USED

3.1 MATERIAL USED:

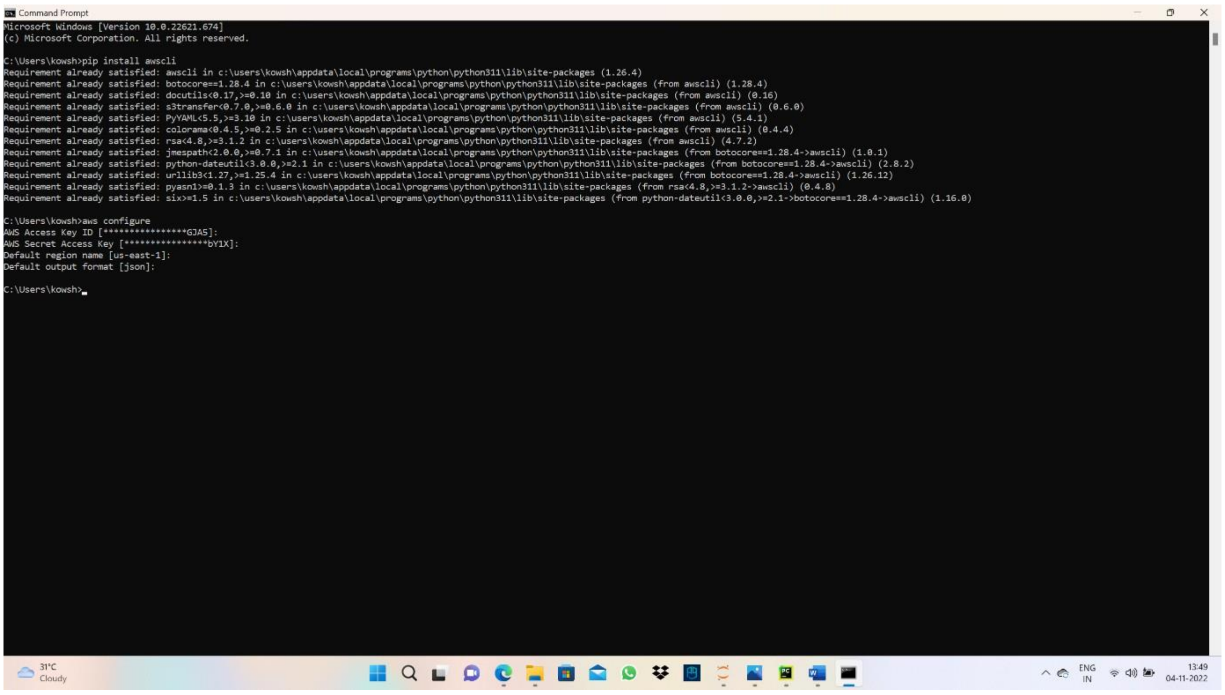
- AWS-CLI (AWS COMMAND LINE INTERFACE)
- IAM (Identify and Access Management)

3.1.1 AWSCLI:

The AWS Command Line Interface (AWS CLI) is a unified tool to manage your AWS services. With just one tool to download and configure, we can control multiple AWS services from the command line and automate them through scripts

3.1.2 IAM:

An AWS Identity and Access Management (IAM) user is an entity that you create in AWS to represent the person application that uses it to interact with AWS.



```
Microsoft Windows [Version 10.0.22621.674]
(c) Microsoft Corporation. All rights reserved.

C:\Users\kowsh>pip install awscli
Requirement already satisfied: awscli in c:\users\kowsh\appdata\local\programs\python\python311\lib\site-packages (1.26.4)
Requirement already satisfied: botocore==1.28.4 in c:\users\kowsh\appdata\local\programs\python\python311\lib\site-packages (from awscli) (1.28.4)
Requirement already satisfied: docutils<0.17,>=0.10 in c:\users\kowsh\appdata\local\programs\python\python311\lib\site-packages (from awscli) (0.16)
Requirement already satisfied: s3transfer<0.7.0,>=0.6.0 in c:\users\kowsh\appdata\local\programs\python\python311\lib\site-packages (from awscli) (0.6.0)
Requirement already satisfied: PyYAML<5.5,>=3.10 in c:\users\kowsh\appdata\local\programs\python\python311\lib\site-packages (from awscli) (5.4.1)
Requirement already satisfied: colorama<0.4.5,>=0.2.5 in c:\users\kowsh\appdata\local\programs\python\python311\lib\site-packages (from awscli) (0.4.4)
Requirement already satisfied: rsa<4.8,>=3.1.2 in c:\users\kowsh\appdata\local\programs\python\python311\lib\site-packages (from awscli) (4.7.2)
Requirement already satisfied: jmespath<2.0.0,>=0.7.1 in c:\users\kowsh\appdata\local\programs\python\python311\lib\site-packages (from botocore==1.28.4->awscli) (1.0.1)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in c:\users\kowsh\appdata\local\programs\python\python311\lib\site-packages (from botocore==1.28.4->awscli) (2.8.2)
Requirement already satisfied: urllib3<1.27,>=1.25.4 in c:\users\kowsh\appdata\local\programs\python\python311\lib\site-packages (from botocore==1.28.4->awscli) (1.26.12)
Requirement already satisfied: pyasn1>=0.1.3 in c:\users\kowsh\appdata\local\programs\python\python311\lib\site-packages (from rsa<4.8,>=3.1.2->awscli) (0.4.8)
Requirement already satisfied: six>=1.5 in c:\users\kowsh\appdata\local\programs\python\python311\lib\site-packages (from python-dateutil<3.0.0,>=2.1->botocore==1.28.4->awscli) (1.16.0)

C:\Users\kowsh>aws configure
AWS Access Key ID [****************GJA5]:
AWS Secret Access Key [****************bY1X]:
Default region name [us-east-1]:
Default output format [json]:

C:\Users\kowsh>
```

Fig 3.1.1-installation of awscli and aws configure

3.2 IMPORTED LIBRARIES:

3.2.1 BOTO3:

Boto3 is the Amazon Web Services (AWS) Software Development Kit (SDK) for Python, which allows Python developers to write software that makes use services like Amazon S3 and Amazon EC2.

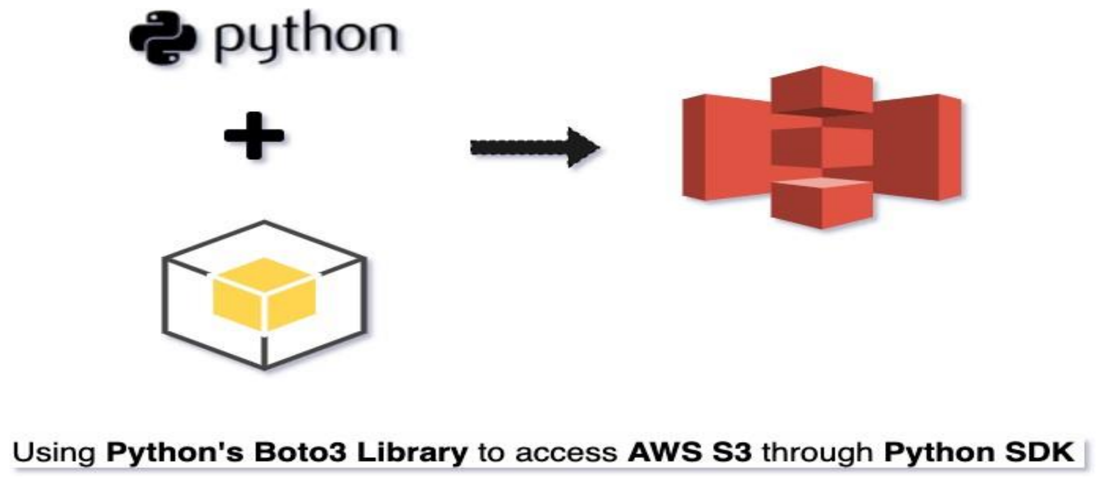

python
+
Using **Python's Boto3 Library** to access **AWS S3** through **Python SDK**

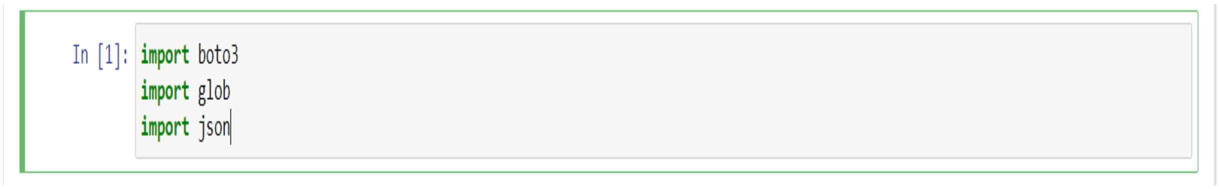
**Fig 3.2.1-boto3

3.2.2 JSON:

Python has a built-in package called json,hich can be used to work with JSON data. Bucket policy options available for granting permission to your Amazon S3 resources. Bucket policy is JSON-based access policy language.

3.2.3 GLOB:

The glob (short for global) is used to return all file paths that match a specific pattern. We can use glob to search for a specific file pattern, or perhaps more usefully, search for files where the filename matches a certain pattern by using wildcard characters

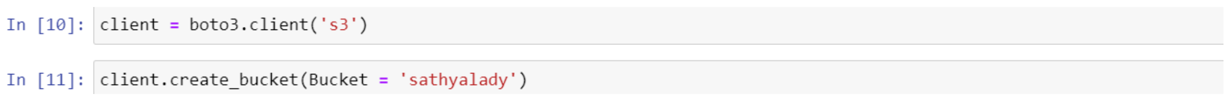


```
In [1]: import boto3
        import glob
        import json
```

Fig 3.2.2-importing libraries

3.3 BUCKET CREATION:

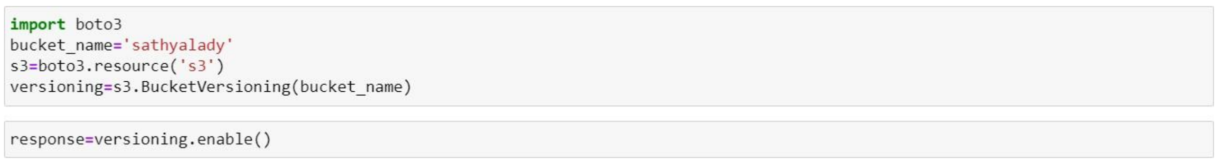
To upload your data (photos, videos, documents, etc.) to Amazon S3, you must first create an S3 bucket in one of the AWS Regions.A bucket is a container for objects stored in Amazon S3. You can store any number of objects in a bucket and can have up to 100 buckets in your account. To request an increase, visit the Service Quotas Console.



```
In [10]: client = boto3.client('s3')

In [11]: client.create_bucket(Bucket = 'sathyalady')
```

Fig 3.3-creating bucket

3.4 VERSIONING: You can use S3 Versioning to keep multiple variants object in the same bucket. With S3 Versioning, you can preserve, retrieve, and restore every version of every object stored in your buckets. You can easily recover from both unintended user actions and application failures.



```
import boto3
bucket_name='sathyalady'
s3=boto3.resource('s3')
versioning=s3.BucketVersioning(bucket_name)

response=versioning.enable()
```

Fig 3.4 – enabling bucket versioning

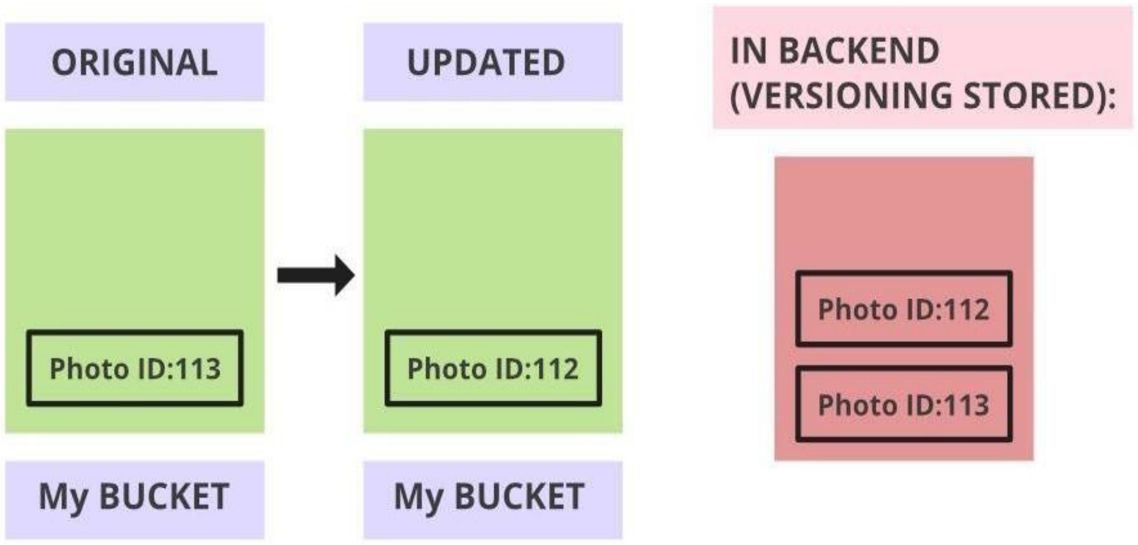
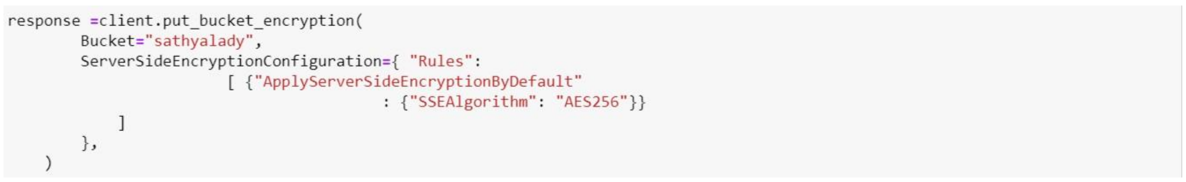


Fig 3.4.1 – how versioning works

3.5 ENCRYPTION:

Amazon S3 supports both server-side encryption (with three key management options: SSE-KMS, SSE-C, SSE-S3) and client-side encryption for data uploads. Amazon S3 offers flexible security features to block unauthorized users from accessing your data. You can set the default encryption behavior on an Amazon S3 bucket so that all objects are encrypted when they are stored in the bucket. The objects are encrypted using server-side encryption with either Amazon S3-managed keys (SSE-S3) or AWS Key Management Service (AWS KMS) keys.

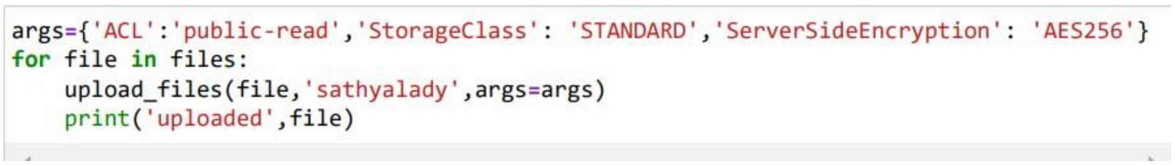


```
response =client.put_bucket_encryption(
        Bucket="sathyalady",
        ServerSideEncryptionConfiguration={ "Rules":
                        [ {"ApplyServerSideEncryptionByDefault"
                                         : {"SSEAlgorithm": "AES256"}}
            ]
        },
    )
```

Fig 3.5 – bucket encryption

3.6 ACL:

Amazon S3 access control lists (ACLs) enable you to manage access to buckets and objects. Each bucket and object has an ACL attached to it as a subresource. It defines which AWS accounts or groups are granted access and the type of access. When a request is received against a resource, Amazon S3 checks the corresponding ACL to verify that the requester has the necessary access permissions.

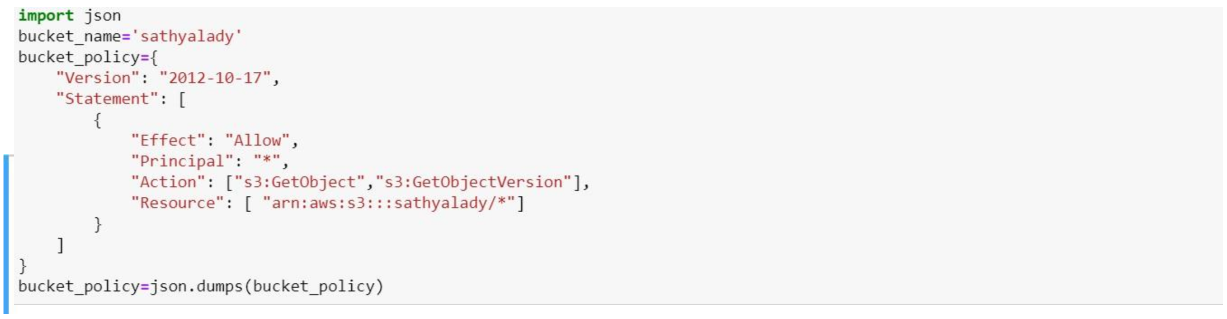


```
args={'ACL':'public-read','StorageClass': 'STANDARD','ServerSideEncryption': 'AES256'}
for file in files:
    upload_files(file,'sathyalady',args=args)
    print('uploaded',file)
```

Fig 3.6 - acl

3.7 BUCKET POLICY:

A bucket policy is a resource-based AWS Identity and Access Management (IAM) policy that you can use to grant access permissions to your bucket and the objects in it. Only the bucket owner can associate a policy with a bucket. The permissions attached to the bucket apply to all of the objects in the bucket that are owned by the bucket owner. Bucket policies are limited to 20 KB in size.Bucket policies use JSON-based access policy language that is standard across AWS. You can use bucket policies to add or deny permissions for the objects in a bucket. Bucket policies allow or deny requests based on the elements in the policy, including the requester, S3 actions, resources, and aspects or conditions of the request (for example, the IP address used to make the request). -



```
import json
bucket_name='sathyalady'
bucket_policy={
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": ["s3:GetObject","s3:GetObjectVersion"],
            "Resource": [ "arn:aws:s3:::sathyalady/*"]
        }
    ]
}
bucket_policy=json.dumps(bucket_policy)
```

Fig 3.7- bucket policy

CHAPTER 4

RESULTS AND DISCUSSION, PERFORMANCEANALYSIS

4.1 MODEL ANALYSIS:

Installation of AWSCLI and adding AWS configuration to system had done successfully. Created a bucket in S3(SIMPLE STORAGE SERVICE) which is provided by AWS Storage. Uploaded files into bucket with some features added to files like encryption, storage class. Added bucket versioning to the bucket which helps to copy the old files when we update a existing file. Encryption is added to the bucket which helps to encrypt the data by not allowing access to unauthorized users to access into our bucket. Because of these types of features provided by the AWS storage many industries and other companies are likely to use AWS Storage. AWS Storage has unlimited storage data that we can store as much data we want and the billing will be done according how much data used.

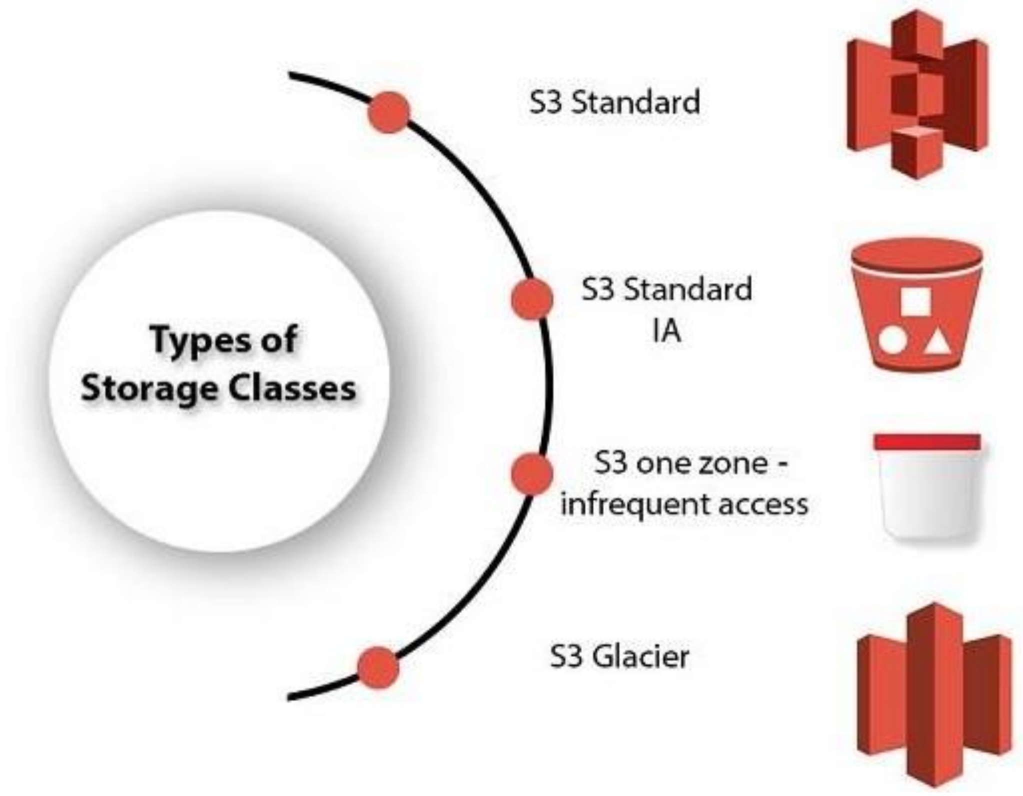


Fig 4.1.1-diff types of storage classes

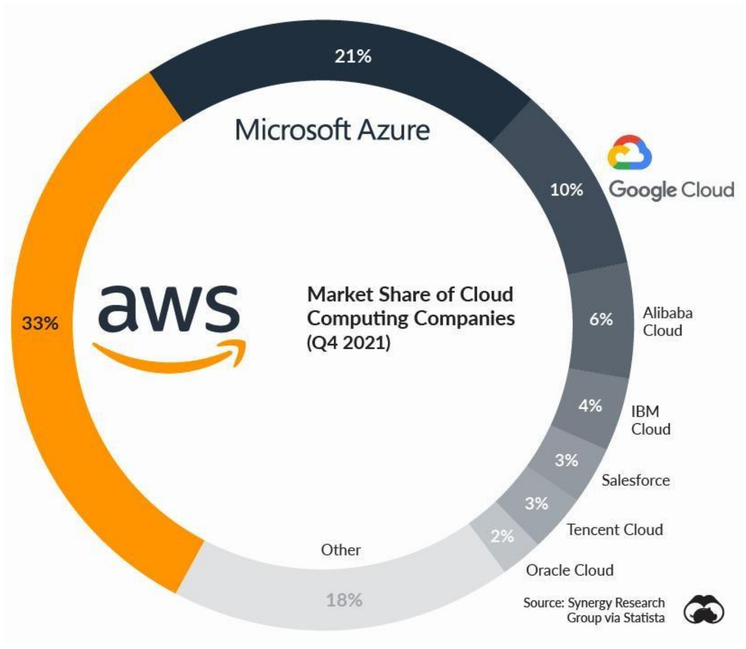


Fig 4.1.2 – pie chart for usage of different cloud services

CHAPTER 5
SUMMARY AND CONCLUSIONS

- Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. You can use Amazon S3 to store and retrieve any amount of data at any time, from anywhere.
- To get the most out of Amazon S3, you need to understand a few simple concepts. Amazon S3 stores data as objects within buckets. An object consists of a file and optionally any metadata that describes that file. To store an object in Amazon S3, you upload the file you want to store to a bucket. When you upload a file, you can set permissions on the object and any metadata.
- Buckets are the containers for objects. You can have one or more buckets. For each bucket, you can control access to it (who can create, delete, and list objects in the bucket), view access logs for it and its objects, and choose the geographical region where Amazon S3 will store the bucket and its contents.

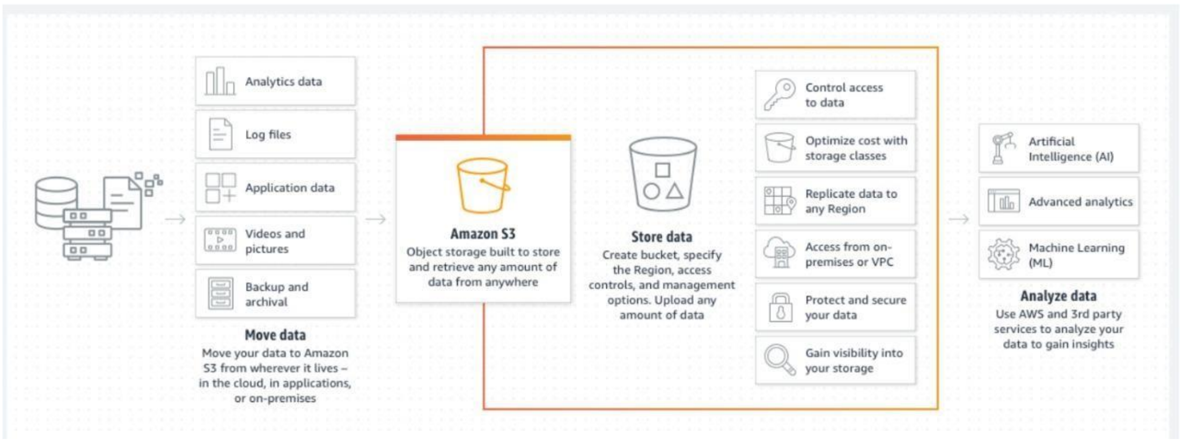

Analytics data
Log files
Application data
Videos and pictures
Backup and archival
Move data
Move your data to Amazon S3 from wherever it lives – in the cloud, in applications, or on-premises
Amazon S3
Object storage built to store and retrieve any amount of data from anywhere
Store data
Create bucket, specify the Region, access controls, and management options. Upload any amount of data
Control access to data
Optimize cost with storage classes
Replicate data to any Region
Access from on-premises or VPC
Protect and secure your data
Gain visibility into your storage
Artificial Intelligence (AI)
Advanced analytics
Machine Learning (ML)
Analyze data
Use AWS and 3rd party services to analyze your data to gain insights

Fig 5.1 – summary of aws

REFERENCES

[1] AWS Console Management

https://us-east-1.console.aws.amazon.com/console/home?region=us-east-1

[2] S3 BOTO3 Documentation

S3 — Boto3 Docs 1.26.2 documentation - Amazon Web Services

[3] Python Language Reference for s3

Python, Boto3, and AWS S3: Demystified – Real Python

[4] For Theory Lectures

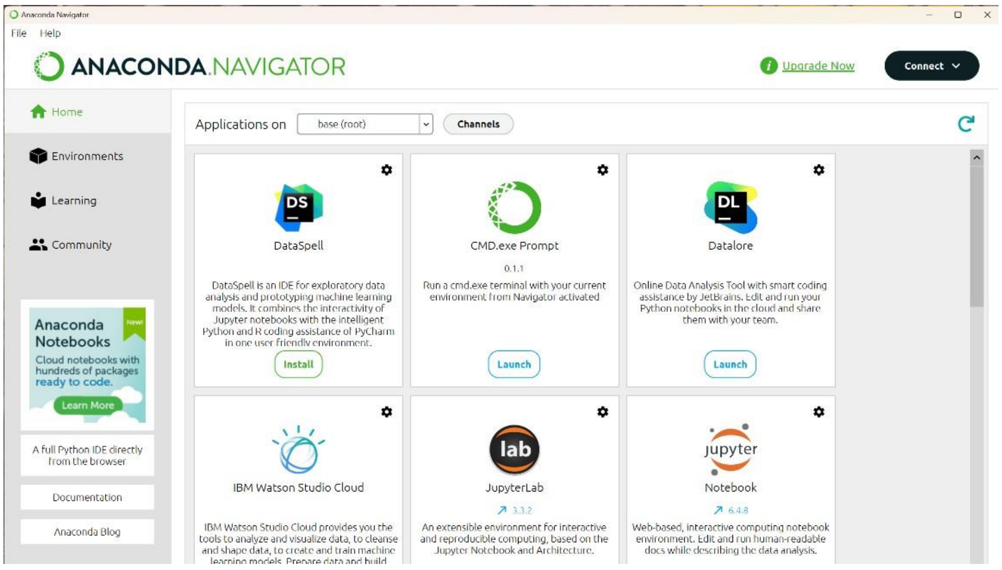
https://drive.google.com/drive/folders/1qkyllkmnTNUgVz_5toPi_9wxE96qHraw?usp=sharing

[5] For Practical Lectures

https://www.youtube.com/watch?v=qsPZL-0OIJg

WORKING ENVIRONMENT

ANACONDA NAVIGATOR is desktop GUI used to launch applications and also manage packages in one place.



DataSpell is an IDE for exploratory data analysis and prototyping machine learning models. It combines the interactivity of Jupyter notebooks with the intelligent Python and R coding assistance of PyCharm in one user friendly environment.

Run a cmd.exe terminal with your current environment from Navigator activated

Online Data Analysis Tool with smart coding assistance by JetBrains. Edit and run your Python notebooks in the cloud and share them with your team.

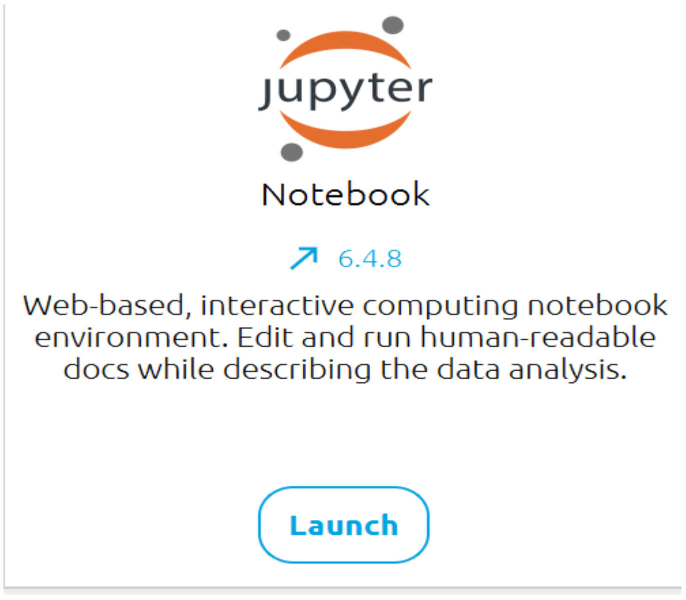
IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

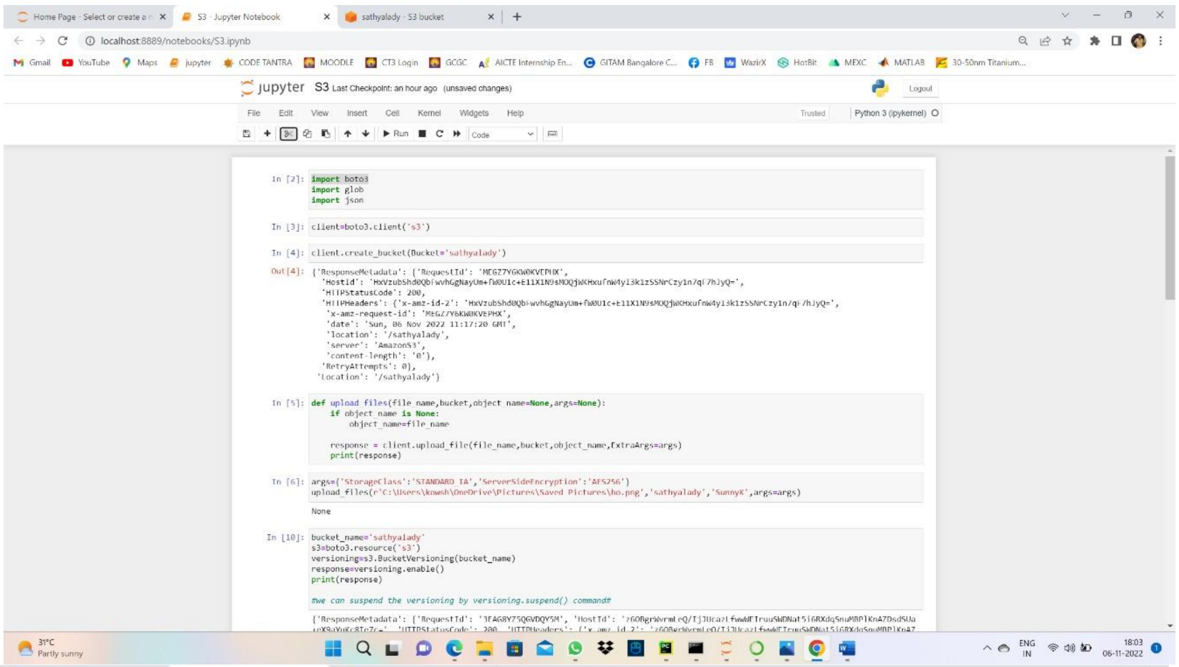
CODING ENVIRONMENT

Jupyter notebook from the anaconda navigator is launched along with all the preinstalled packages for python.

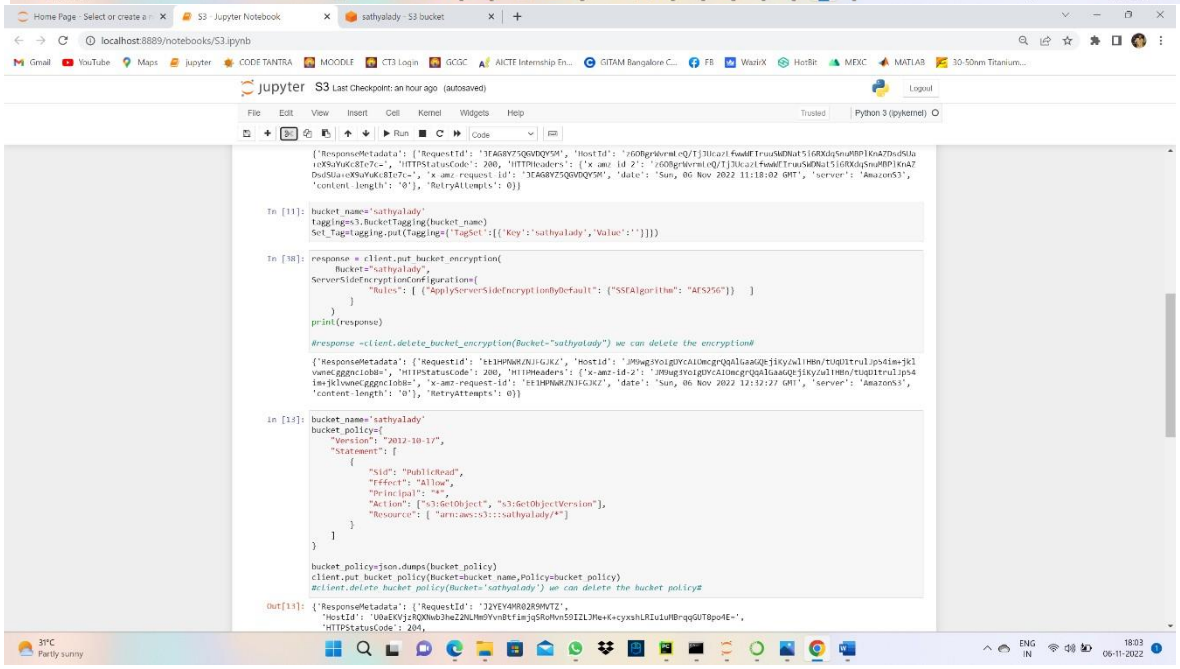


Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

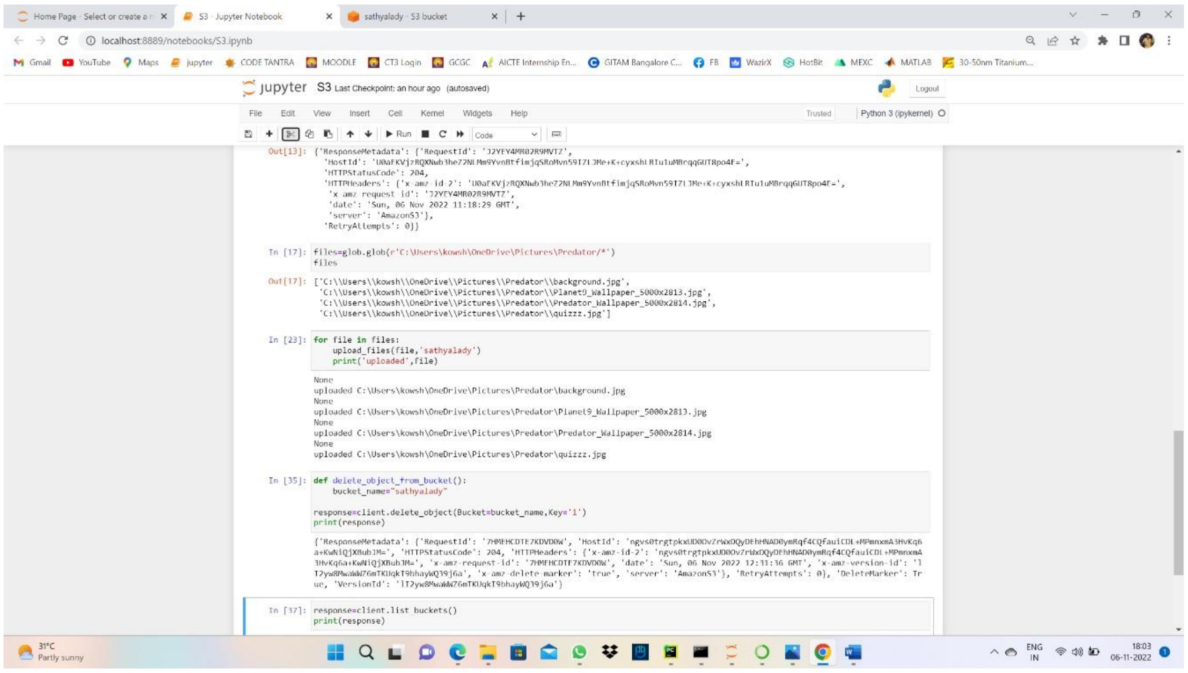
SCREENSHOTS AND OUTPUTS:



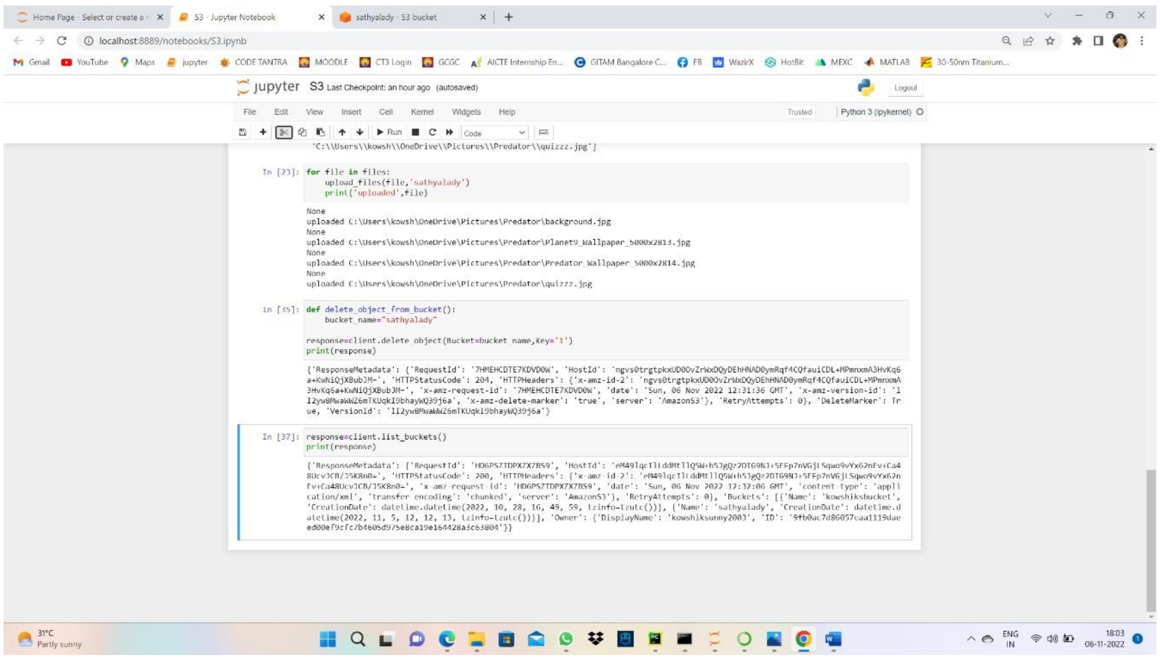
```
In [2]: import boto3
        import glob
        import json

In [3]: client=boto3.client('s3')

In [4]: client.create_bucket(Bucket='sathyalady')

Out[4]: {'ResponseMetadata': {'RequestId': 'MEGZ7YGKW0KVEPHX',
          'HostId': 'HxVzubShd0QbFwvhGgNayUm+fW0U1c+E11X1N9sMOQjWKHxufnW4yI3k1zSSNrCzy1n7qF7hJyQ=',
          'HTTPStatusCode': 200,
          'HTTPHeaders': {'x-amz-id-2': 'HxVzubShd0QbFwvhGgNayUm+fW0U1c+E11X1N9sMOQjWKHxufnW4yI3k1zSSNrCzy1n7qF7hJyQ=',
           'x-amz-request-id': 'MEGZ7Y6KW0KVEPHX',
           'date': 'Sun, 06 Nov 2022 11:17:20 GMT',
           'location': '/sathyalady',
           'server': 'AmazonS3',
           'content-length': '0'},
          'RetryAttempts': 0},
         'Location': '/sathyalady'}

In [5]: def upload_files(file_name,bucket,object_name=None,args=None):
            if object_name is None:
                object_name=file_name

            response = client.upload_file(file_name,bucket,object_name,ExtraArgs=args)
            print(response)

In [6]: args={'StorageClass':'STANDARD_IA','ServerSideEncryption':'AES256'}
        upload_files(r'C:\Users\kowsh\OneDrive\Pictures\Saved Pictures\ho.png','sathyalady','SunnyK',args=args)

        None

In [10]: bucket_name='sathyalady'
         s3=boto3.resource('s3')
         versioning=s3.BucketVersioning(bucket_name)
         response=versioning.enable()
         print(response)

         #we can suspend the versioning by versioning.suspend() command#

         {'ResponseMetadata': {'RequestId': '3EAG8YZ5QGVDQY5M', 'HostId': 'z6OBgrWvrmLeQ/TjJUcazLfwwWETruuSWDNat5i6RXdqSnuMBPlKnAZDsdSUa
```



```
{'ResponseMetadata': {'RequestId': '3EAG8YZ5QGVDQY5M', 'HostId': 'z6OBgrWvrmLeQ/TjJUcazLfwwWETruuSWDNat5i6RXdqSnuMBPlKnAZDsdSUa
+eX9aYuKc8Te7c=', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amz-id-2': 'z6OBgrWvrmLeQ/TjJUcazLfwwWETruuSWDNat5i6RXdqSnuMBPlKnAZ
DsdSUa+eX9aYuKc8Te7c=', 'x-amz-request-id': '3EAG8YZ5QGVDQY5M', 'date': 'Sun, 06 Nov 2022 11:18:02 GMT', 'server': 'AmazonS3',
'content-length': '0'}, 'RetryAttempts': 0}}

In [11]: bucket_name='sathyalady'
         tagging=s3.BucketTagging(bucket_name)
         Set_Tag=tagging.put(Tagging={'TagSet':[{'Key':'sathyalady','Value':''}]})

In [38]: response = client.put_bucket_encryption(
              Bucket="sathyalady",
         ServerSideEncryptionConfiguration={
                   "Rules": [ {"ApplyServerSideEncryptionByDefault": {"SSEAlgorithm": "AES256"}}   ]
               }
             )
         print(response)

         #response =client.delete_bucket_encryption(Bucket="sathyalady") we can delete the encryption#

         {'ResponseMetadata': {'RequestId': 'EE1HPNWRZNJFGJKZ', 'HostId': 'JM9wg3YoIgDYcAIOmcgrQqAlGaaGQEjiKyZwlTHBn/tUqD1trulJp54im+jkl
         vwneCgggncIob8=', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amz-id-2': 'JM9wg3YoIgDYcAIOmcgrQqAlGaaGQEjiKyZwlTHBn/tUqD1trulJp54
         im+jklvwneCgggncIob8=', 'x-amz-request-id': 'EE1HPNWRZNJFGJKZ', 'date': 'Sun, 06 Nov 2022 12:32:27 GMT', 'server': 'AmazonS3',
         'content-length': '0'}, 'RetryAttempts': 0}}

In [13]: bucket_name='sathyalady'
         bucket_policy={
             "Version": "2012-10-17",
             "Statement": [
                 {
                     "Sid": "PublicRead",
                     "Effect": "Allow",
                     "Principal": "*",
                     "Action": ["s3:GetObject", "s3:GetObjectVersion"],
                     "Resource": [ "arn:aws:s3:::sathyalady/*"]
                 }
             ]
         }

         bucket_policy=json.dumps(bucket_policy)
         client.put_bucket_policy(Bucket=bucket_name,Policy=bucket_policy)
         #client.delete_bucket_policy(Bucket='sathyalady') we can delete the bucket policy#

Out[13]: {'ResponseMetadata': {'RequestId': 'J2YEY4MR02R9MVTZ',
           'HostId': 'U0aEKVjzRQXNwb3heZ2NLMm9YvnBtfimjqSRoMvn59IZLJMe+K+cyxshLRIu1uMBrqqGUT8po4E=',
           'HTTPStatusCode': 204,
```

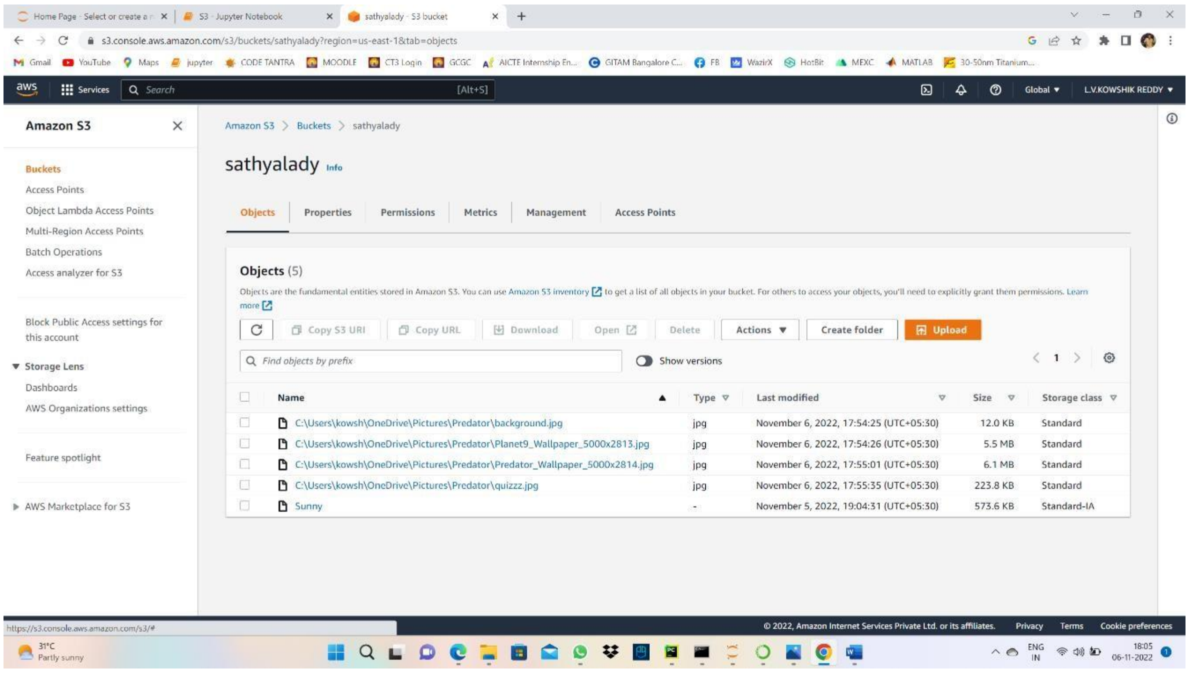


```
Out[13]: {'ResponseMetadata': {'RequestId': 'J2YEY4MR02R9MVTZ',
           'HostId': 'U0aFKVjzRQXNwb3heZ2NLMm9YvnBtfimjqSRoMvn59IZLJMe+K+cyxshLRTu1uMBrqqGUT8po4F=',
           'HTTPStatusCode': 204,
           'HTTPHeaders': {'x-amz-id-2': 'U0aFKVjzRQXNwb3heZ2NLMm9YvnBtfimjqSRoMvn59IZLJMe+K+cyxshLRTu1uMBrqqGUT8po4F=',
            'x-amz-request-id': 'J2YEY4MR02R9MVTZ',
            'date': 'Sun, 06 Nov 2022 11:18:29 GMT',
            'server': 'AmazonS3'},
           'RetryAttempts': 0}}

In [17]: files=glob.glob(r'C:\Users\kowsh\OneDrive\Pictures\Predator/*')
         files

Out[17]: ['C:\\Users\\kowsh\\OneDrive\\Pictures\\Predator\\background.jpg',
          'C:\\Users\\kowsh\\OneDrive\\Pictures\\Predator\\Planet9_Wallpaper_5000x2813.jpg',
          'C:\\Users\\kowsh\\OneDrive\\Pictures\\Predator\\Predator_Wallpaper_5000x2814.jpg',
          'C:\\Users\\kowsh\\OneDrive\\Pictures\\Predator\\quizzz.jpg']

In [23]: for file in files:
             upload_files(file,'sathyalady')
             print('uploaded',file)

         None
         uploaded C:\Users\kowsh\OneDrive\Pictures\Predator\background.jpg
         None
         uploaded C:\Users\kowsh\OneDrive\Pictures\Predator\Planet9_Wallpaper_5000x2813.jpg
         None
         uploaded C:\Users\kowsh\OneDrive\Pictures\Predator\Predator_Wallpaper_5000x2814.jpg
         None
         uploaded C:\Users\kowsh\OneDrive\Pictures\Predator\quizzz.jpg

In [35]: def delete_object_from_bucket():
             bucket_name="sathyalady"

         response=client.delete_object(Bucket=bucket_name,Key='1')
         print(response)

         {'ResponseMetadata': {'RequestId': '7HMEHCDTE7KDVD0W', 'HostId': 'ngvs0trgtpkxUD0OvZrWxDQyDEhHNAD0ymRqf4CQfauiCDL+MPmnxmA3HvKq6
         a+KwNiQjXBubJM=', 'HTTPStatusCode': 204, 'HTTPHeaders': {'x-amz-id-2': 'ngvs0trgtpkxUD0OvZrWxDQyDEhHNAD0ymRqf4CQfauiCDL+MPmnxmA
         3HvKq6a+KwNiQjXBubJM=', 'x-amz-request-id': '7HMEHCDTE7KDVD0W', 'date': 'Sun, 06 Nov 2022 12:31:36 GMT', 'x-amz-version-id': 'l
         I2yw8MwaWWZ6mTKUqkI9bhayWQ39j6a', 'x-amz-delete-marker': 'true', 'server': 'AmazonS3'}, 'RetryAttempts': 0}, 'DeleteMarker': Tr
         ue, 'VersionId': 'lI2yw8MwaWWZ6mTKUqkI9bhayWQ39j6a'}

In [37]: response=client.list_buckets()
         print(response)
```



```
          'C:\\Users\\kowsh\\OneDrive\\Pictures\\Predator\\quizzz.jpg']

In [23]: for file in files:
             upload_files(file,'sathyalady')
             print('uploaded',file)

         None
         uploaded C:\Users\kowsh\OneDrive\Pictures\Predator\background.jpg
         None
         uploaded C:\Users\kowsh\OneDrive\Pictures\Predator\Planet9_Wallpaper_5000x2813.jpg
         None
         uploaded C:\Users\kowsh\OneDrive\Pictures\Predator\Predator_Wallpaper_5000x2814.jpg
         None
         uploaded C:\Users\kowsh\OneDrive\Pictures\Predator\quizzz.jpg

In [35]: def delete_object_from_bucket():
             bucket_name="sathyalady"

         response=client.delete_object(Bucket=bucket_name,Key='1')
         print(response)

         {'ResponseMetadata': {'RequestId': '7HMEHCDTE7KDVD0W', 'HostId': 'ngvs0trgtpkxUD0OvZrWxDQyDEhHNAD0ymRqf4CQfauiCDL+MPmnxmA3HvKq6
         a+KwNiQjXBubJM-', 'HTTPStatusCode': 204, 'HTTPHeaders': {'x-amz-id-2': 'ngvs0trgtpkxUD0OvZrWxDQyDEhHNAD0ymRqf4CQfauiCDL+MPmnxmA
         3HvKq6a+KwNiQjXBubJM-', 'x-amz-request-id': '7HMEHCDTE7KDVD0W', 'date': 'Sun, 06 Nov 2022 12:31:36 GMT', 'x-amz-version-id': 'l
         I2yw8MwaWWZ6mTKUqkI9bhayWQ39j6a', 'x-amz-delete-marker': 'true', 'server': 'AmazonS3'}, 'RetryAttempts': 0}, 'DeleteMarker': Tr
         ue, 'VersionId': 'lI2yw8MwaWWZ6mTKUqkI9bhayWQ39j6a'}

In [37]: response=client.list_buckets()
         print(response)
```

```
         {'ResponseMetadata': {'RequestId': 'HD6PSZTDPX7X7BS9', 'HostId': 'eM49lqcIlLddMtllQ5W+h5JgQz2DTG9NJ+5FFp7nVGjLSqwo9vYx62nFv+Ca4
         8UcvJCB/J5K8n0=', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amz-id-2': 'eM49lqcIlLddMtllQ5W+h5JgQz2DTG9NJ+5FFp7nVGjLSqwo9vYx62n
         Fv+Ca48UcvJCB/J5K8n0=', 'x-amz-request-id': 'HD6PSZTDPX7X7BS9', 'date': 'Sun, 06 Nov 2022 12:32:06 GMT', 'content-type': 'appli
         cation/xml', 'transfer-encoding': 'chunked', 'server': 'AmazonS3'}, 'RetryAttempts': 0}, 'Buckets': [{'Name': 'kowshiksbucket',
         'CreationDate': datetime.datetime(2022, 10, 28, 16, 49, 59, tzinfo=tzutc())}, {'Name': 'sathyalady', 'CreationDate': datetime.d
         atetime(2022, 11, 5, 12, 12, 13, tzinfo=tzutc())}], 'Owner': {'DisplayName': 'kowshiksunny2003', 'ID': '9fb0ac7d86057caa1119dae
         ed00ef9cfc7b4605d975e8ca19e164428a3c63804'}}
```


s3.console.aws.amazon.com/s3/buckets/sathyalady?region=us-east-1&tab=objects

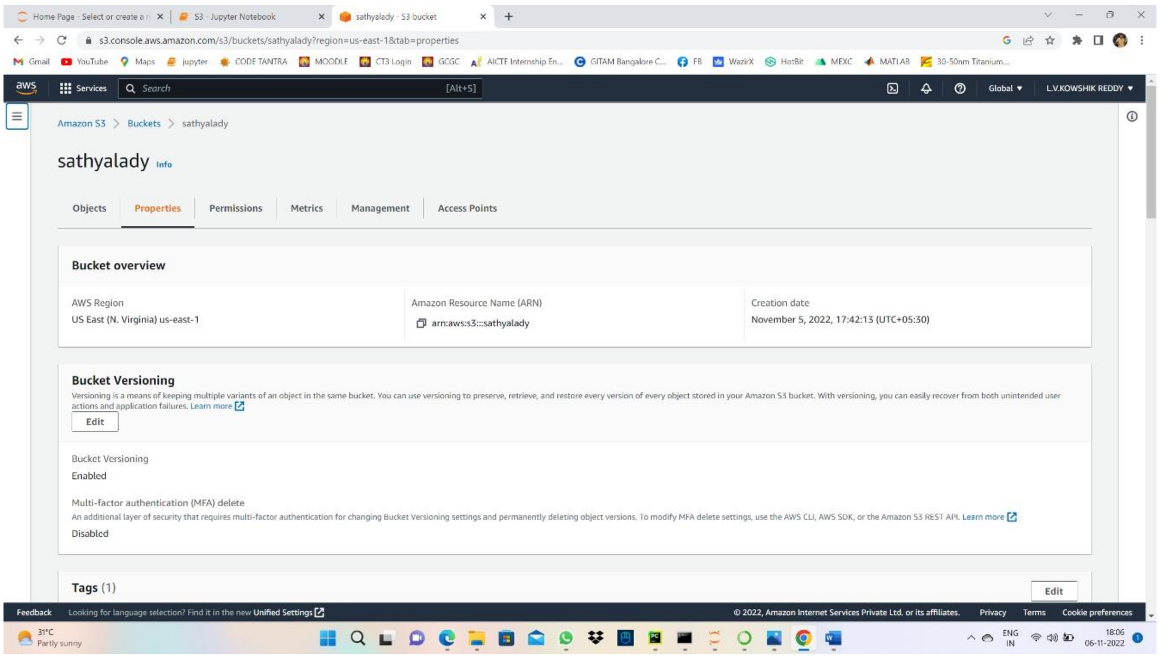
Amazon S3 > Buckets > sathyalady

sathyalady

Objects (5)

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
C:\Users\kowsh\OneDrive\Pictures\Predator\background.jpg	jpg	November 6, 2022, 17:54:25 (UTC+05:30)	12.0 KB	Standard
C:\Users\kowsh\OneDrive\Pictures\Predator\Planet9_Wallpaper_5000x2813.jpg	jpg	November 6, 2022, 17:54:26 (UTC+05:30)	5.5 MB	Standard
C:\Users\kowsh\OneDrive\Pictures\Predator\Predator_Wallpaper_5000x2814.jpg	jpg	November 6, 2022, 17:55:01 (UTC+05:30)	6.1 MB	Standard
C:\Users\kowsh\OneDrive\Pictures\Predator\quizzz.jpg	jpg	November 6, 2022, 17:55:35 (UTC+05:30)	223.8 KB	Standard
Sunny	-	November 5, 2022, 19:04:31 (UTC+05:30)	573.6 KB	Standard-IA


s3.console.aws.amazon.com/s3/buckets/sathyalady?region=us-east-1&tab=properties

Amazon S3 > Buckets > sathyalady

sathyalady

Bucket overview

AWS Region	Amazon Resource Name (ARN)	Creation date
US East (N. Virginia) us-east-1	arn:aws:s3:::sathyalady	November 5, 2022, 17:42:13 (UTC+05:30)

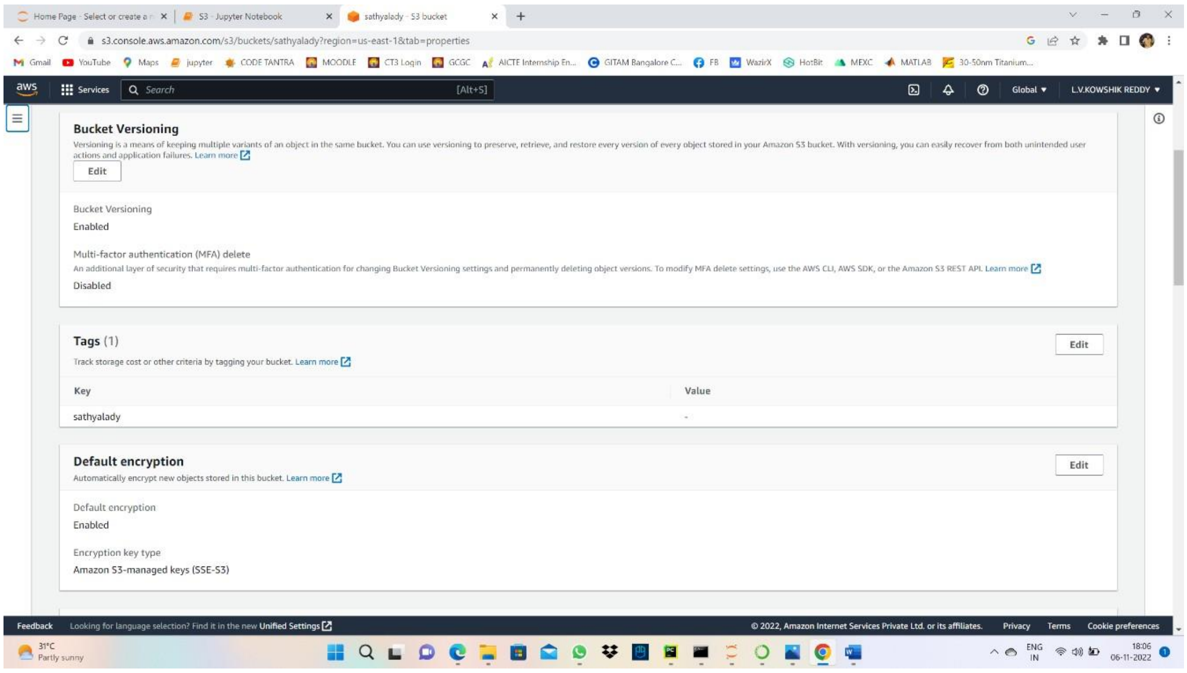
Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. Learn more

Bucket Versioning
Enabled

Multi-factor authentication (MFA) delete
An additional layer of security that requires multi-factor authentication for changing Bucket Versioning settings and permanently deleting object versions. To modify MFA delete settings, use the AWS CLI, AWS SDK, or the Amazon S3 REST API. Learn more
Disabled

Tags (1)


Home Page · Select or create a
S3 · Jupyter Notebook
sathyalady · S3 bucket
s3.console.aws.amazon.com/s3/buckets/sathyalady?region=us-east-1&tab=properties
Services
Search
[Alt+S]
Global
L.V.KOWSHIK REDDY

Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. Learn more

Edit

Bucket Versioning
Enabled

Multi-factor authentication (MFA) delete
An additional layer of security that requires multi-factor authentication for changing Bucket Versioning settings and permanently deleting object versions. To modify MFA delete settings, use the AWS CLI, AWS SDK, or the Amazon S3 REST API. Learn more
Disabled

Tags (1)

Track storage cost or other criteria by tagging your bucket. Learn more

Edit

Key	Value
sathyalady	-

Default encryption

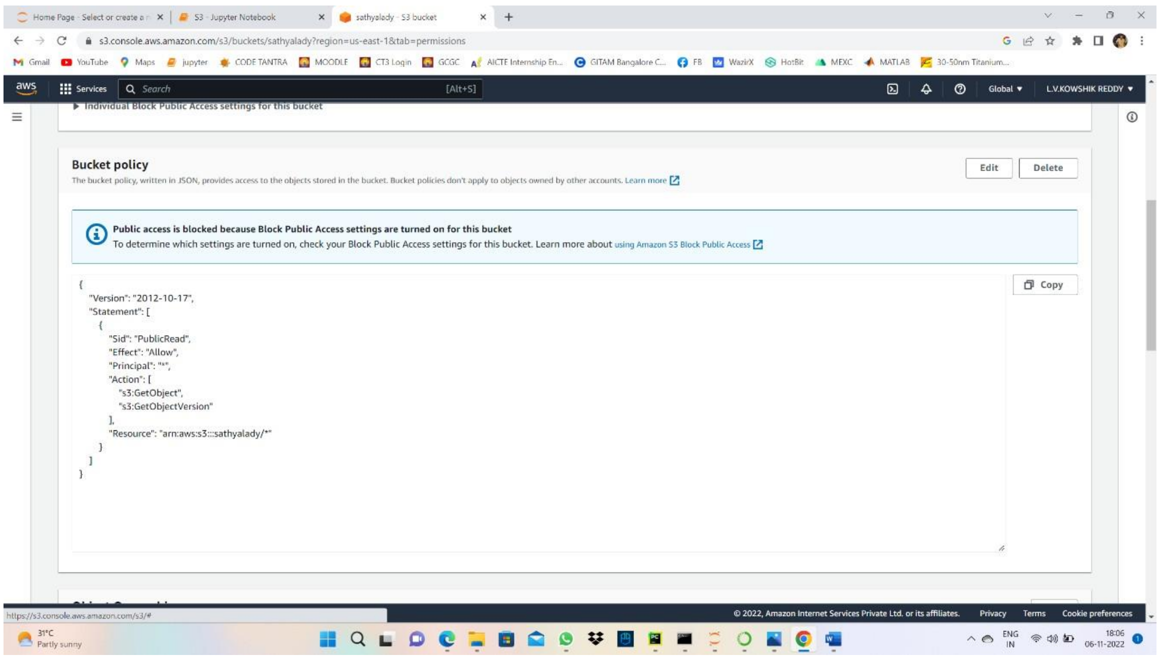
Automatically encrypt new objects stored in this bucket. Learn more

Edit

Default encryption
Enabled

Encryption key type
Amazon S3-managed keys (SSE-S3)

Feedback
Looking for language selection? Find it in the new Unified Settings
© 2022, Amazon Internet Services Private Ltd. or its affiliates.
Privacy
Terms
Cookie preferences
31°C
Partly sunny
ENG
IN
18:06
06-11-2022


Home Page · Select or create a
S3 · Jupyter Notebook
sathyalady · S3 bucket
s3.console.aws.amazon.com/s3/buckets/sathyalady?region=us-east-1&tab=permissions
Services
Search
[Alt+S]
Global
L.V.KOWSHIK REDDY
▶ Individual Block Public Access settings for this bucket

Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more

Edit Delete

Public access is blocked because Block Public Access settings are turned on for this bucket
To determine which settings are turned on, check your Block Public Access settings for this bucket. Learn more about using Amazon S3 Block Public Access

Copy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicRead",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": "arn:aws:s3:::sathyalady/*"
    }
  ]
}
```

https://s3.console.aws.amazon.com/s3/#
© 2022, Amazon Internet Services Private Ltd. or its affiliates.
Privacy
Terms
Cookie preferences
31°C
Partly sunny
ENG
IN
18:06
06-11-2022

SOURCE CODE

```
import boto3
import glob
import json
client=boto3.client('s3')
client.create_bucket(Bucket='sathyalady')
def upload_files(file_name,bucket,object_name=None,args=None):
    if object_name is None:
        object_name=file_name

    response = client.upload_file(file_name,bucket,object_name,ExtraArgs=args)
    print(response)
args={'StorageClass':'STANDARD_IA','ServerSideEncryption':'AES256'}
upload_files(r'C:\Users\kowsh\OneDrive\Pictures\Saved
Pictures\ho.png','sathyalady','SunnyK',args=args)
bucket_name='sathyalady'
s3=boto3.resource('s3')
versioning=s3.BucketVersioning(bucket_name)
response=versioning.enable()
print(response)

#we can suspend the versioning by versioning.suspend() command#
bucket_name='sathyalady'
tagging=s3.BucketTagging(bucket_name)
Set_Tag=tagging.put(Tagging={'TagSet':[{'Key':'sathyalady','Value':''}]})
response = client.put_bucket_encryption(
     Bucket="sathyalady",
ServerSideEncryptionConfiguration={
            "Rules": [ {"ApplyServerSideEncryptionByDefault": {"SSEAlgorithm":
"AES256"}}   ]
        }
    )
print(response)
```

```
#response =client.delete_bucket_encryption(Bucket="sathyalady") we can delete the
encryption#
bucket_name='sathyalady'
bucket_policy={
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicRead",
            "Effect": "Allow",
            "Principal": "*",
            "Action": ["s3:GetObject", "s3:GetObjectVersion"],
            "Resource": [ "arn:aws:s3:::sathyalady/*"]
        }
    ]
}
bucket_policy=json.dumps(bucket_policy)
client.put_bucket_policy(Bucket=bucket_name,Policy=bucket_policy)
#client.delete_bucket_policy(Bucket='sathyalady') we can delete the bucket policy#
files=glob.glob(r'C:\Users\kowsh\OneDrive\Pictures\Predator/*')
files
for file in files:
    upload_files(file,'sathyalady')
    print('uploaded',file)
def delete_object_from_bucket():
    bucket_name="sathyalady"

response=client.delete_object(Bucket=bucket_name,Key='1')
print(response)
response=client.list_buckets()
print(response)
```

------THE END-----