geyser data

User Guide

Getting Started with Geyser Data

Your complete guide to creating your account and managing your cold data cloud archive.

www.GeyserData.com

Getting Started with Geyser Data

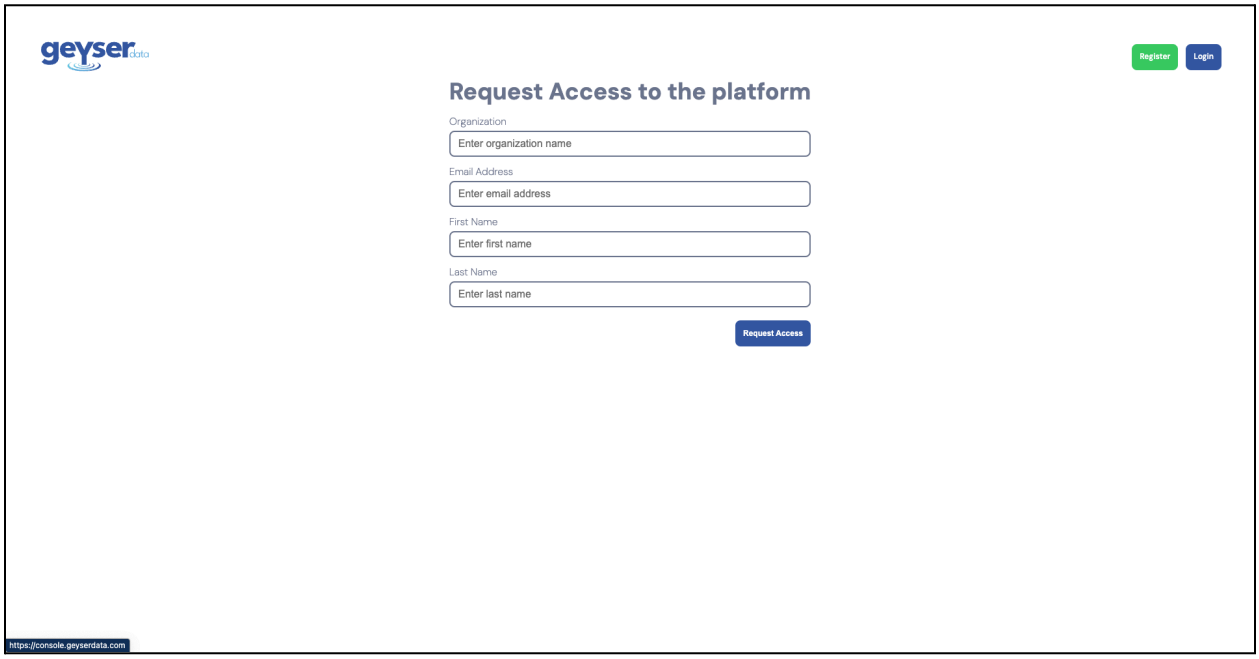
v25.11.26

Create an account 1
Activate & Login 4
Create a Cloud Tape Library 8
View and Manage Your Cloud Tape Library 10
Create Access and Secret Keys 12
Write Data to Your Cloud Tape Library 13
Browse Data in Your Cloud Tape Library 16
Read Data from Your Cloud Tape Library 17
Air Gap Feature 21
Tape Collections 21
Useful links 23
- How to use Cyberduck with Geyser data 23
- S3 Browser Setup Guide 23
- Veeam Integration Guide 23
- Cohesity Integration Guide 23
- MinIO Integration Guide 23

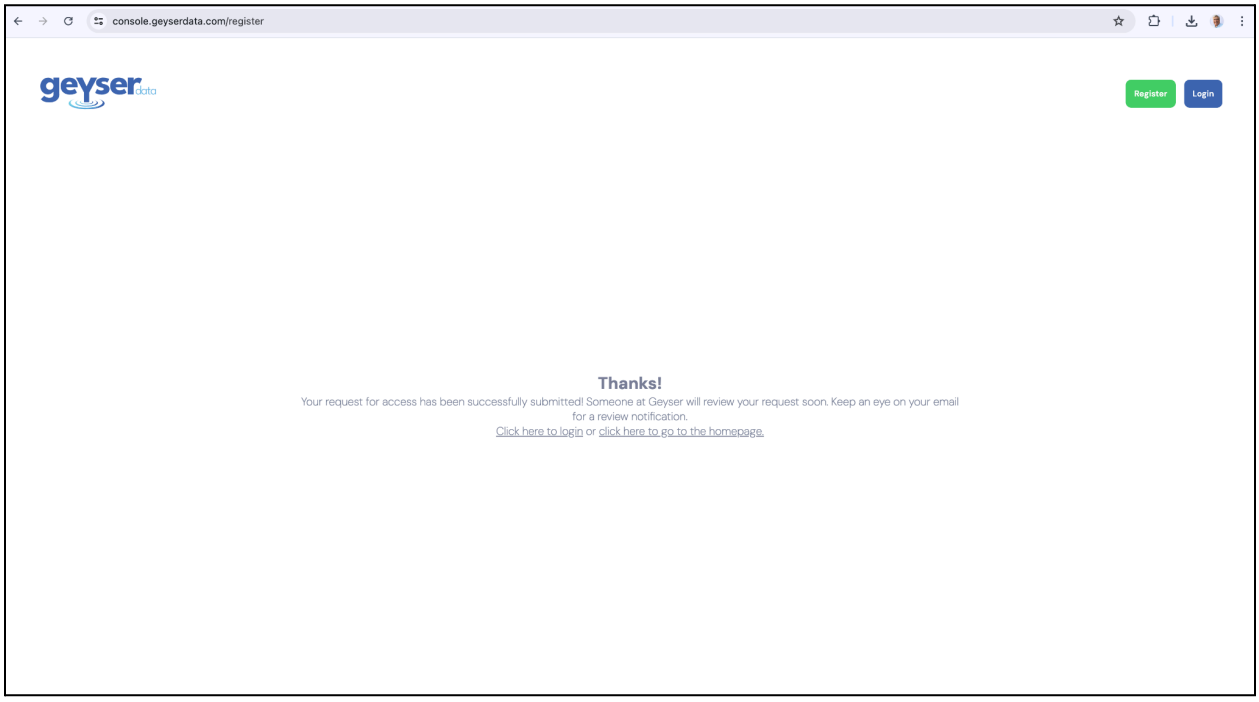
Support 23

Create an account

- Go to https://console.geyserdata.com
- Click **Register** button (top right)
- Enter you Organization Name, Email Address, First Name and Last Name
- Click **Request Access**


Register
Login
Request Access to the platform
Organization
Enter organization name
Email Address
Enter email address
First Name
Enter first name
Last Name
Enter last name
Request Access
https://console.geyserdata.com

- You will get a thank you screen.

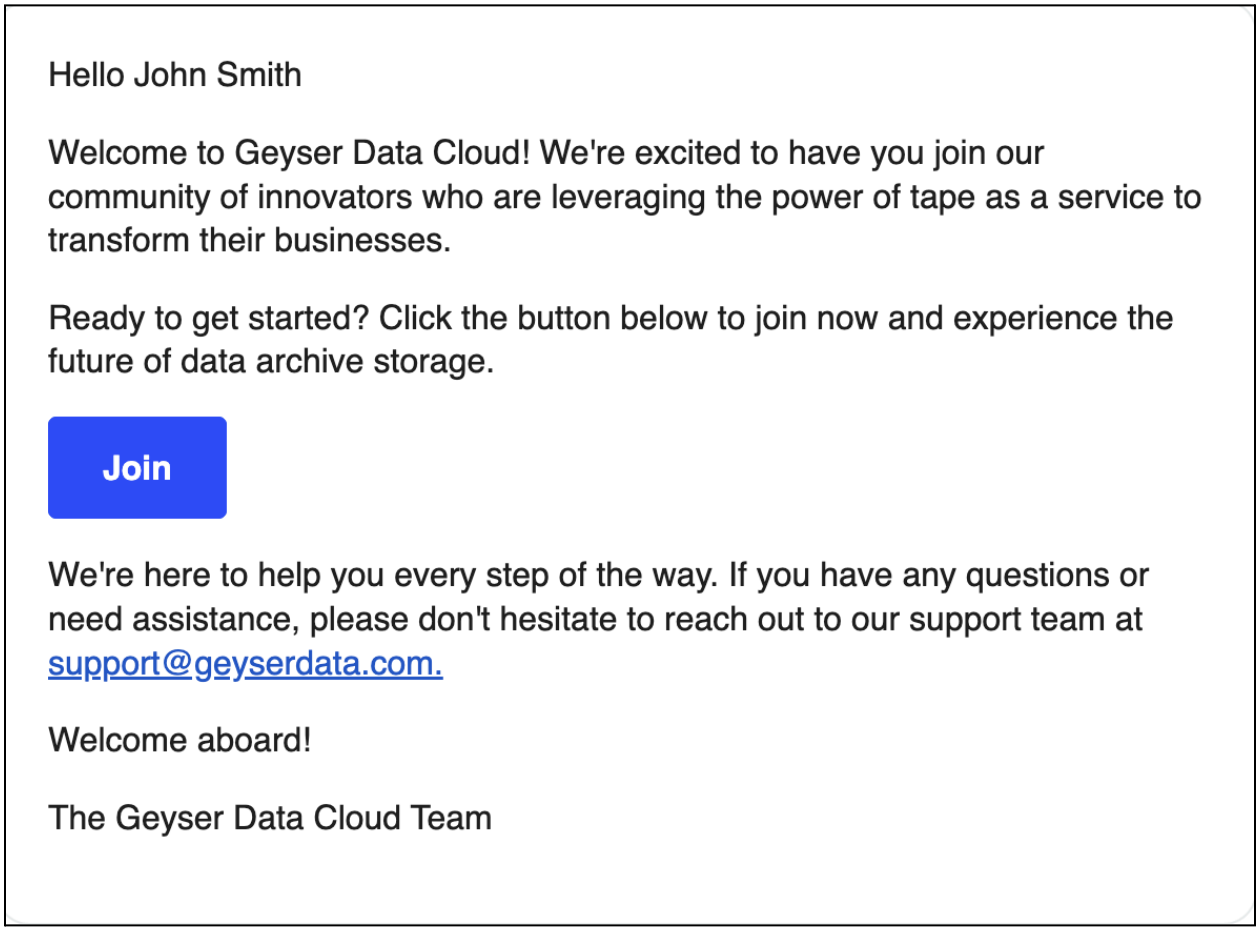


Your request will be reviewed (typically within 24 hours)

If you don't receive a Welcome email within 24 hours, please check your spam folder or email to support@geyserdata.com

Activate & Login

- Click **Join** in your Welcome Email to set a password (8-100 characters).



Hello John Smith

Welcome to Geyser Data Cloud! We're excited to have you join our community of innovators who are leveraging the power of tape as a service to transform their businesses.

Ready to get started? Click the button below to join now and experience the future of data archive storage.

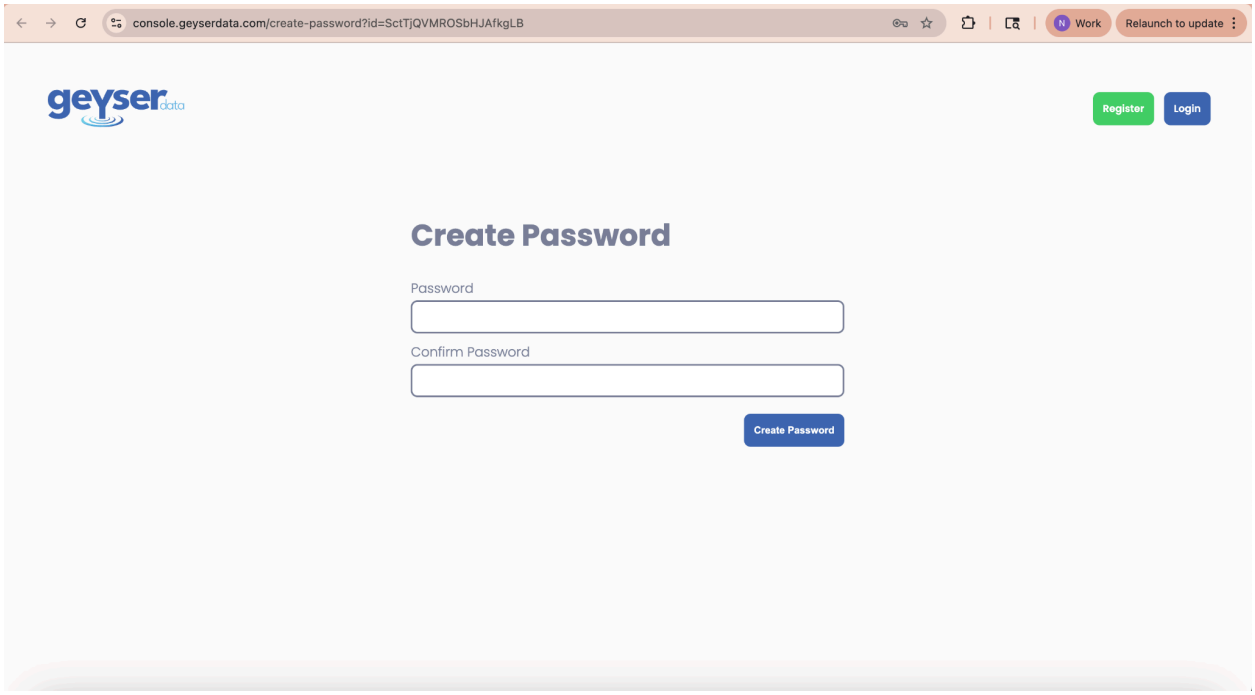
Join

We're here to help you every step of the way. If you have any questions or need assistance, please don't hesitate to reach out to our support team at support@geyserdata.com.

Welcome aboard!

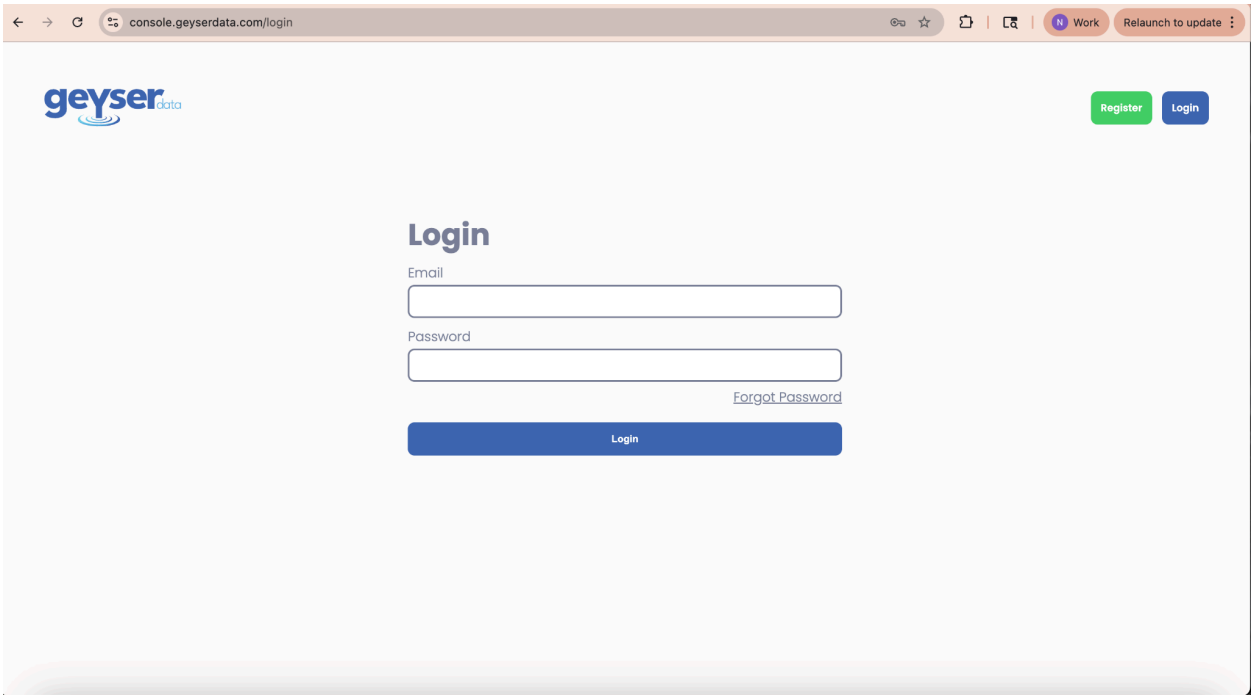
The Geyser Data Cloud Team

- You will be redirected to the Create Password page

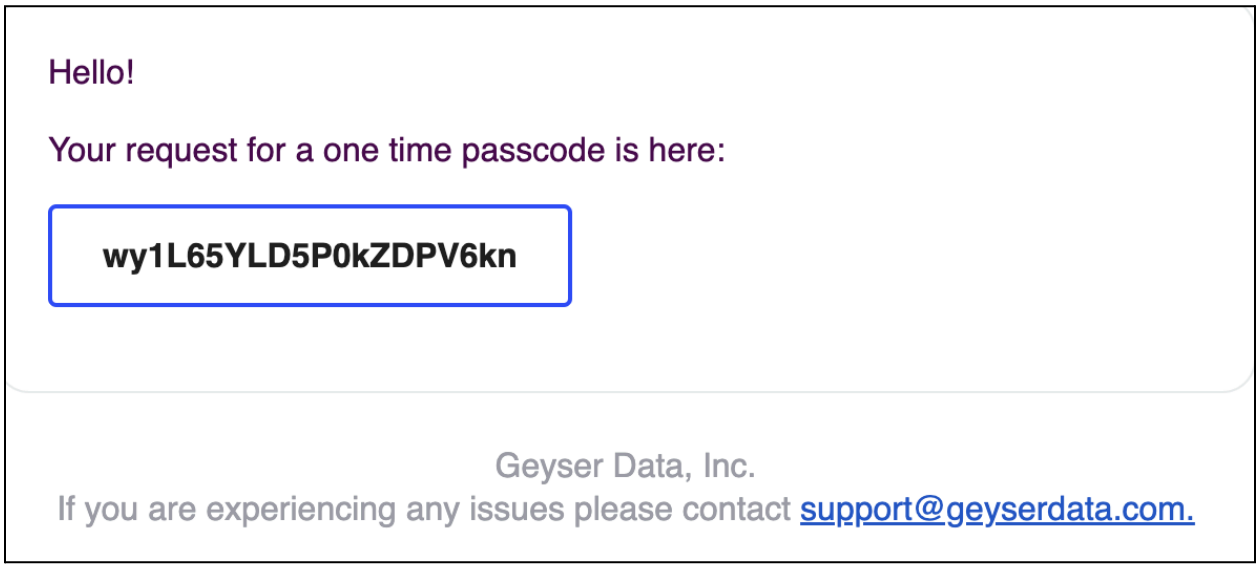


After setting your password you can go ahead and log in!

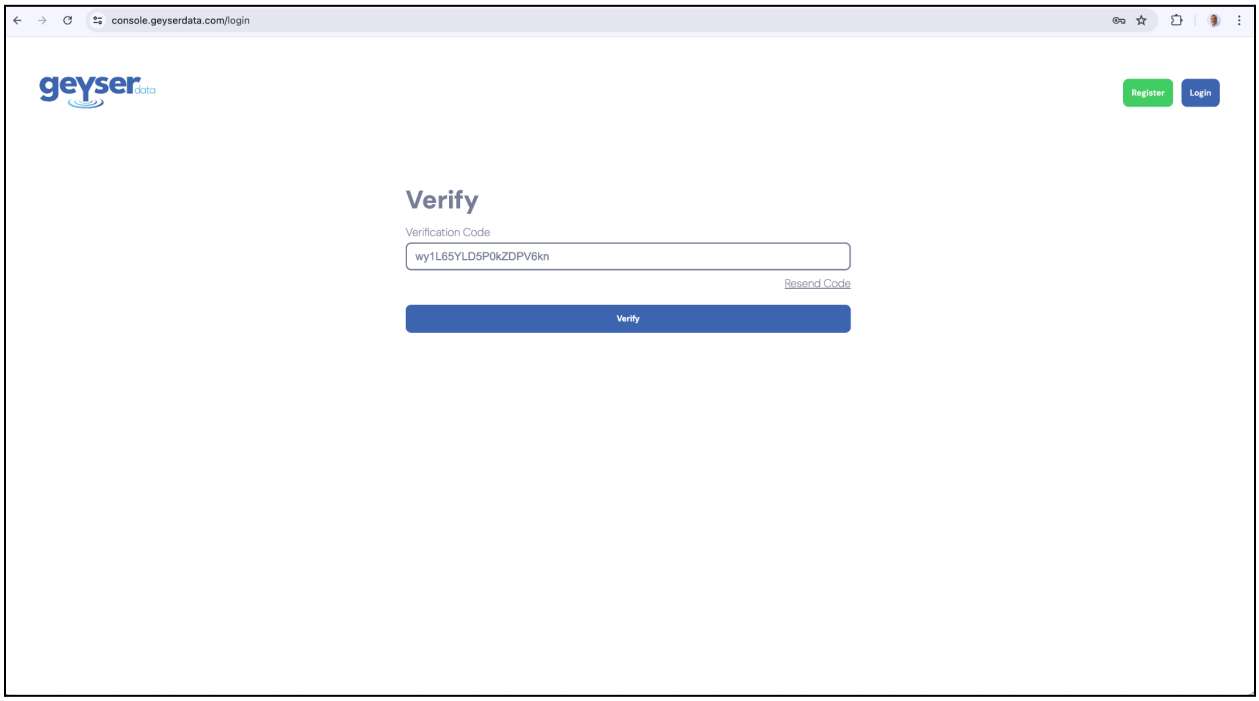
- Enter your email address and password, then click **Login.**



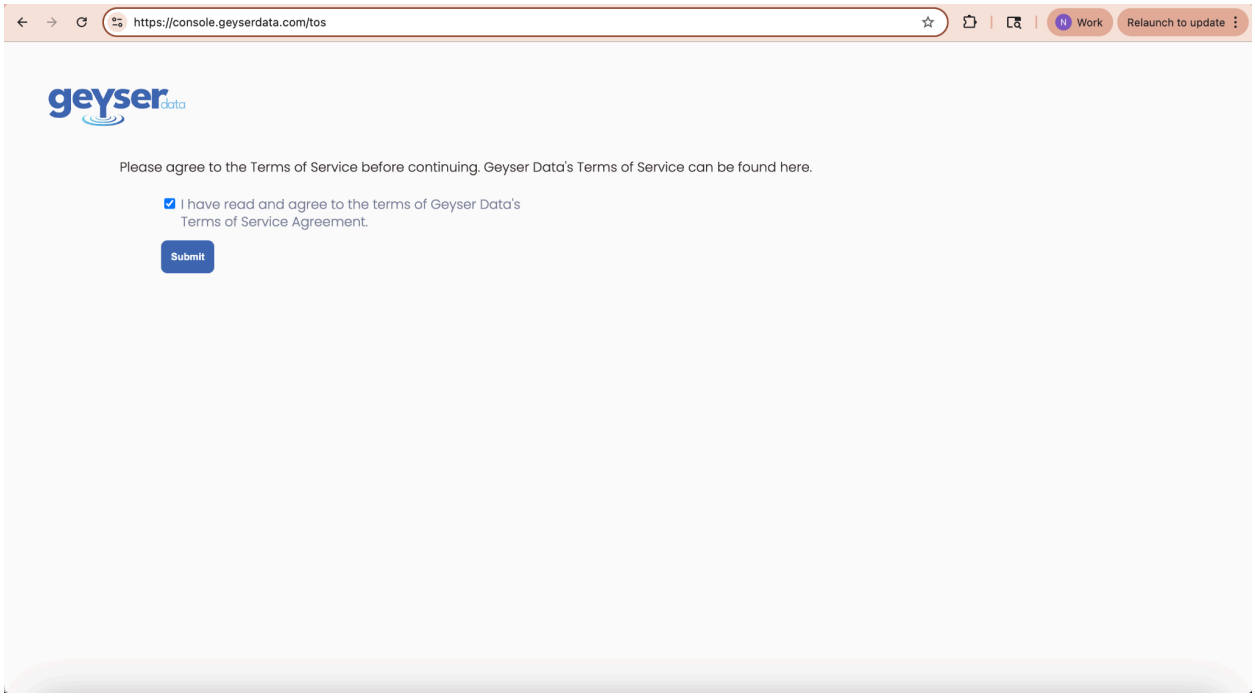
- Enter the **verification code** sent to your email (two-factor authentication).



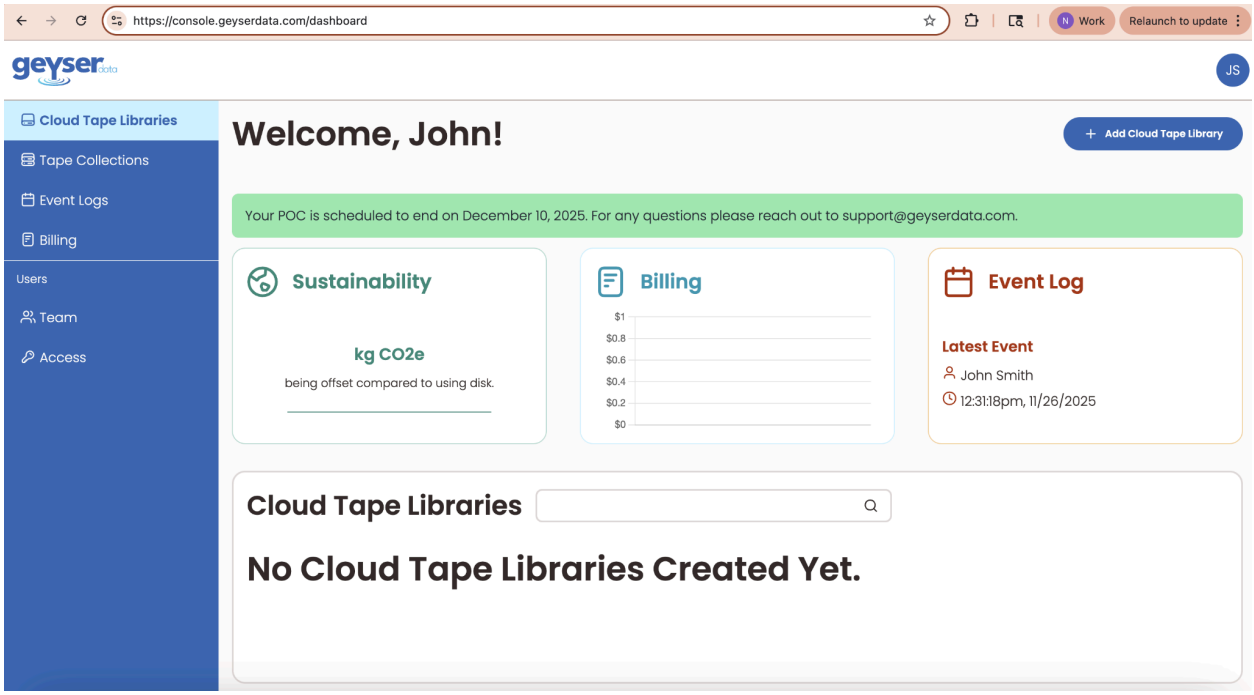
- Copy the **verification code** from the email, paste into the Verify screen and click **Verify**.



- Finally, agree to our terms of service.

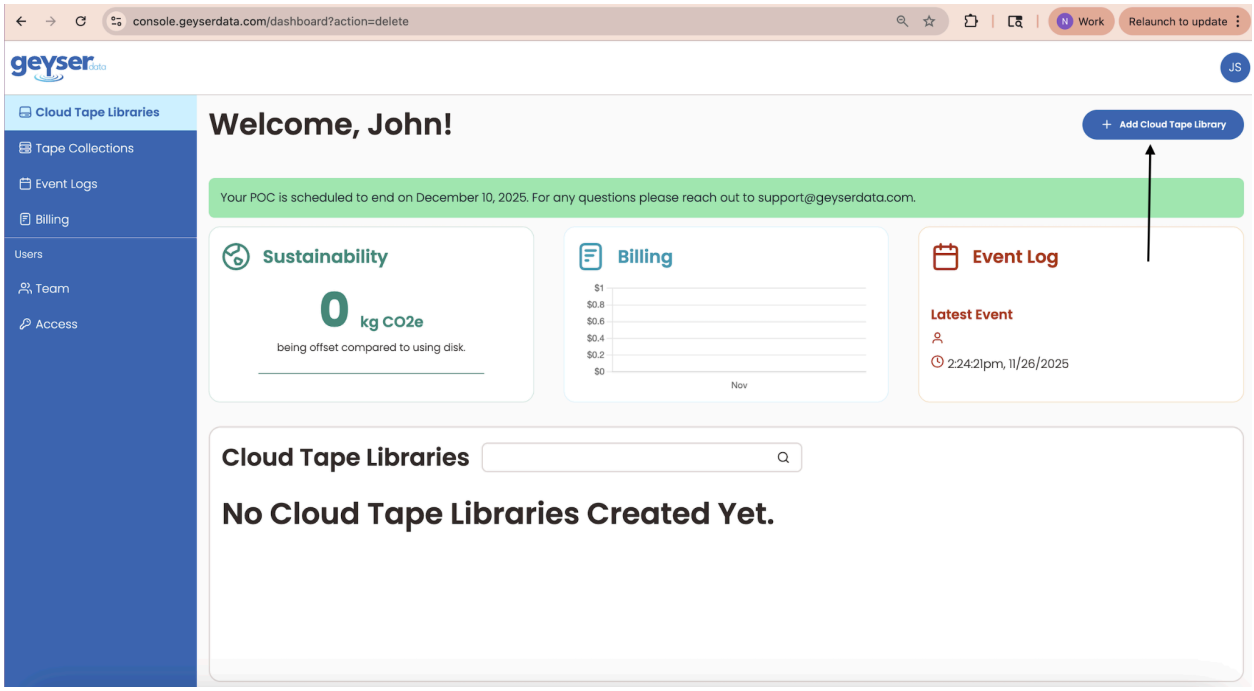


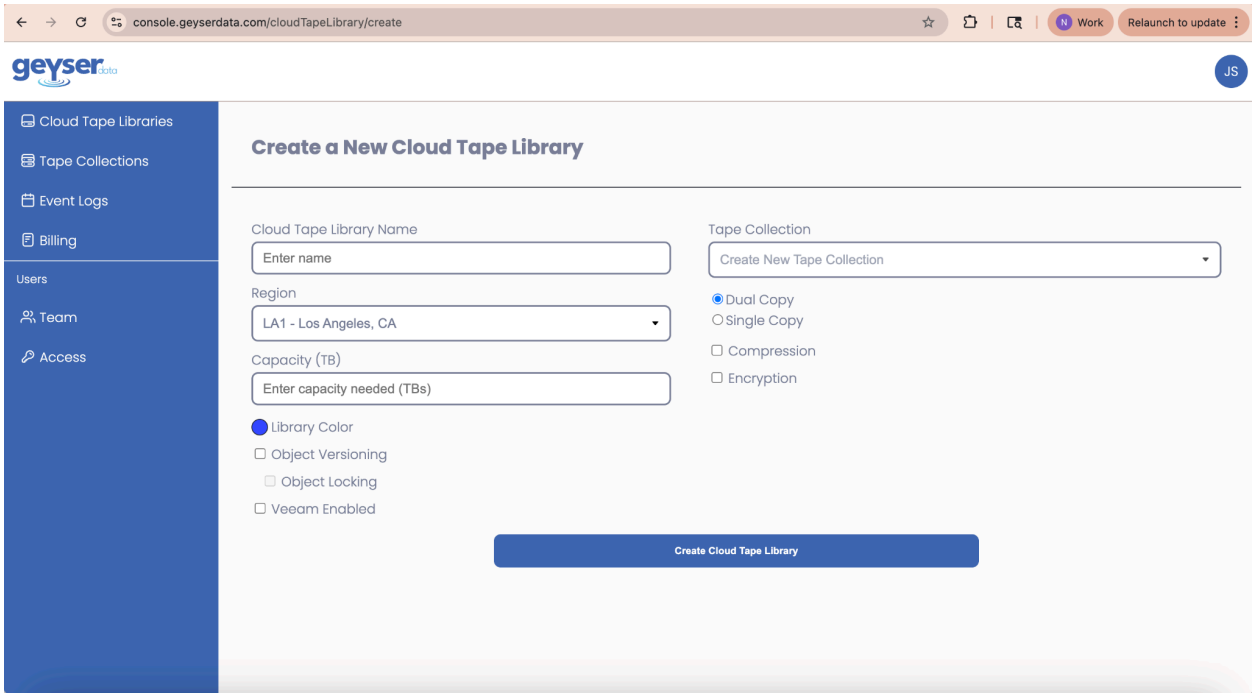
- Congratulations! You are now on your Geyser Data dashboard.



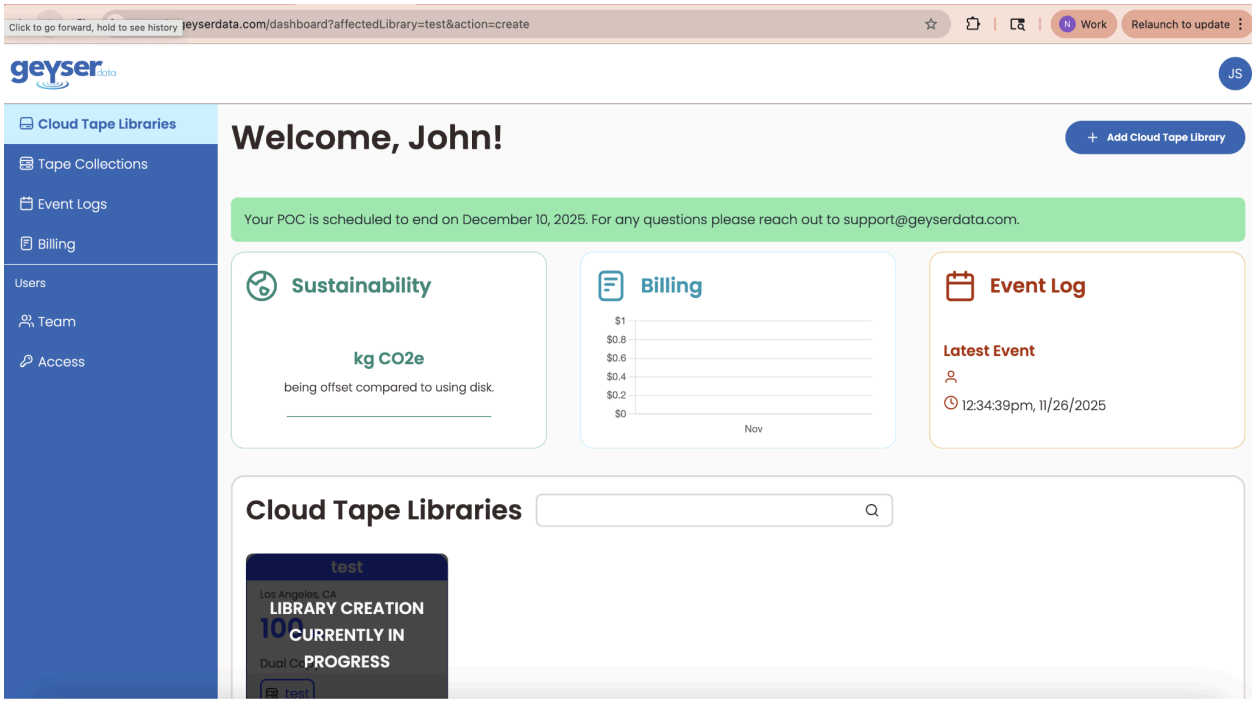
Create a Cloud Tape Library

- On the dashboard, click on **+ Add Cloud Tape Library** (top right corner)



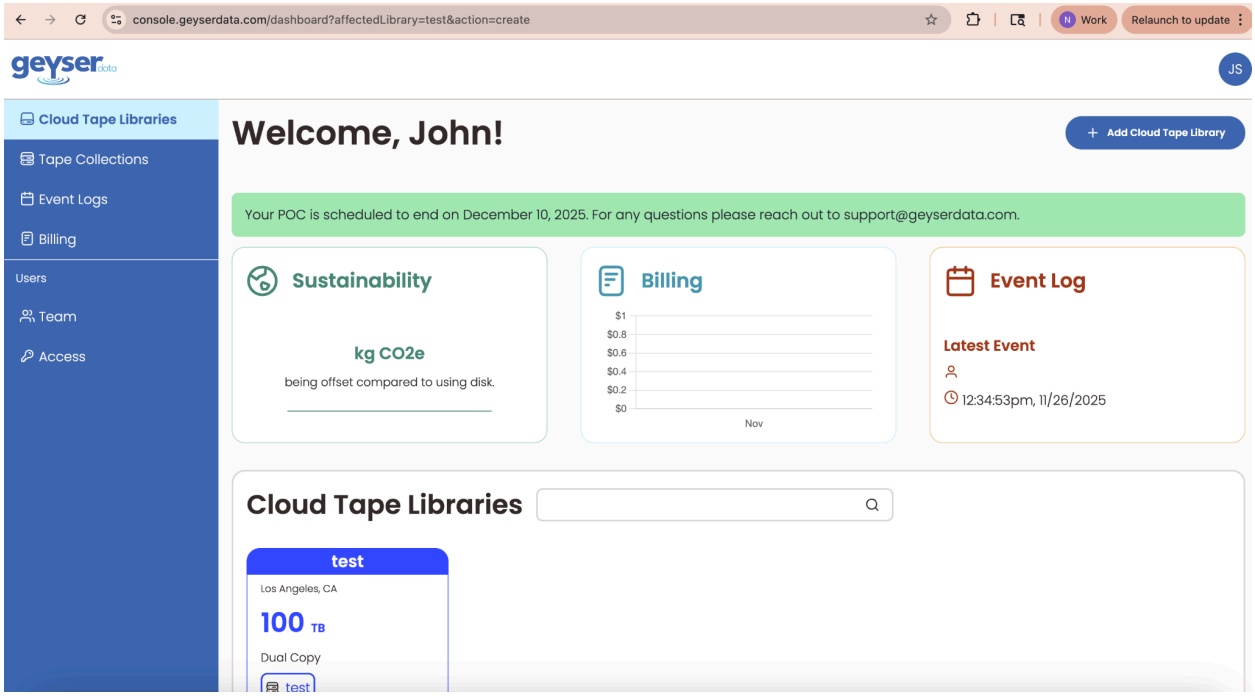


- Fill out the form:
    - Enter Cloud Tape Library Name
    - Choose Region from the dropdown.
    - Provisioned Capacity in TBs. (The capacity rounds up to the next full Tape)
    - Color: Choose a color to visually organize your libraries.
    - Object Versioning: Enabling the bucket to retain multiple copies with the same file name and path.
    - Object Locking: Enabling a lock on any object so it cannot be deleted.
    - Veeam Enabled: Enabling this bucket for the Veeam Archive tier.
    - Tape Collection: Default to Create a new tape collection (see Tape Collections section for more details)
    - Copy Type: Select Single Copy or Dual Copy. (For production environments we recommend dual copy, ie the data is written to two tapes
    - Optional: Enable Compression, Encryption (These are available at additional cost)
- Click **Create Cloud Tape Library**
- Watch for the “Library Creation Currently in Progress” message on the dashboard - setup takes less than 1 minute.

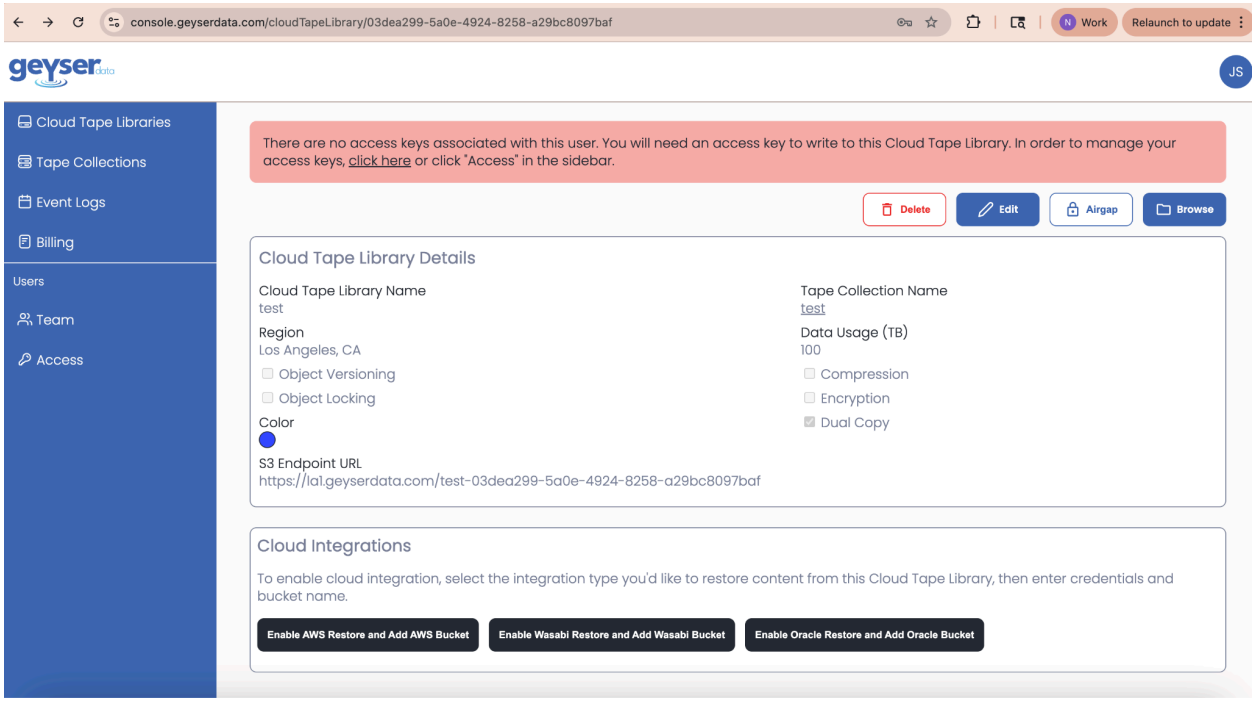


View and Manage Your Cloud Tape Library

- Once the Cloud Tape Library is created, it will show up as a tile on the Dashboard screen.



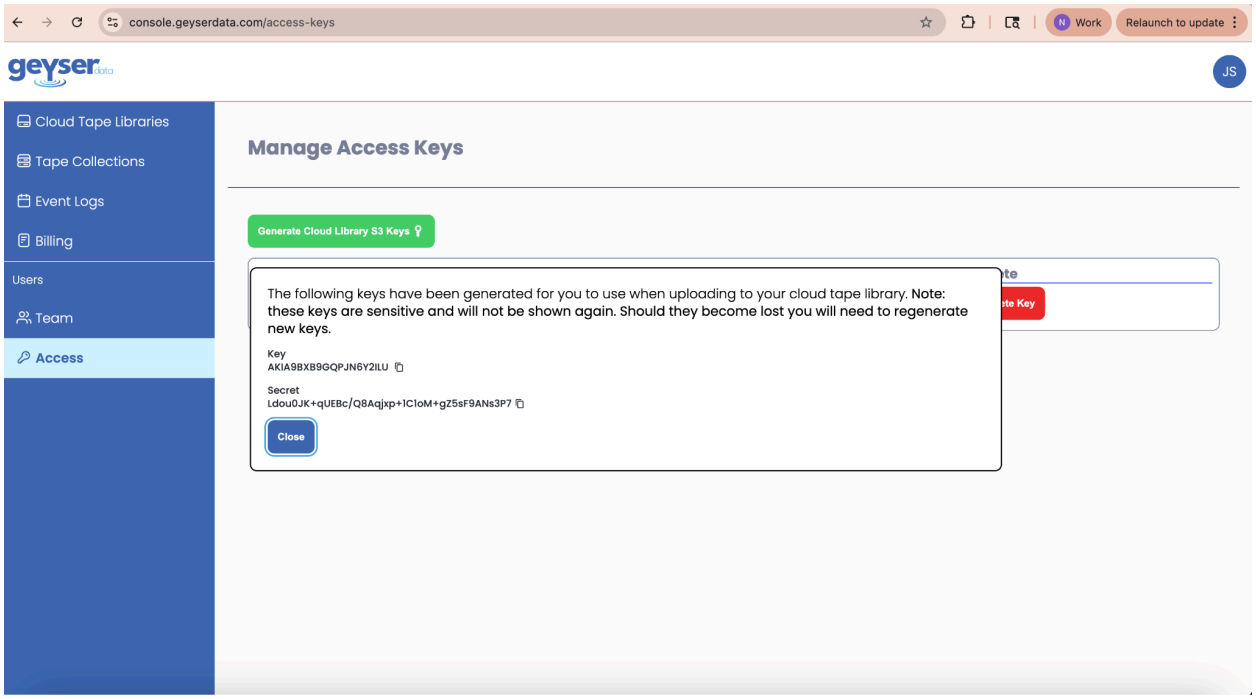
- Click on the **Cloud Tape Library tile** to access the View Cloud Tape Library screen. This screen provides a comprehensive overview of your library:
    - Information about the Cloud Tape Library
    - Tape Collection settings
    - s3Url - the s3 endpoint used for directing S3 commands through CLI/API or application integrations



- If you see a red banner, you need to generate Access/Secret keys first.

Create Access and Secret Keys

- If prompted, click **Click Here** in the red banner or click **Access** (left panel)
- Click **Generate Cloud Library S3 keys**
- Copy BOTH Access and Secret Keys to a password manager or secure file.
  For security reasons, you cannot view these keys again—regenerate as needed.



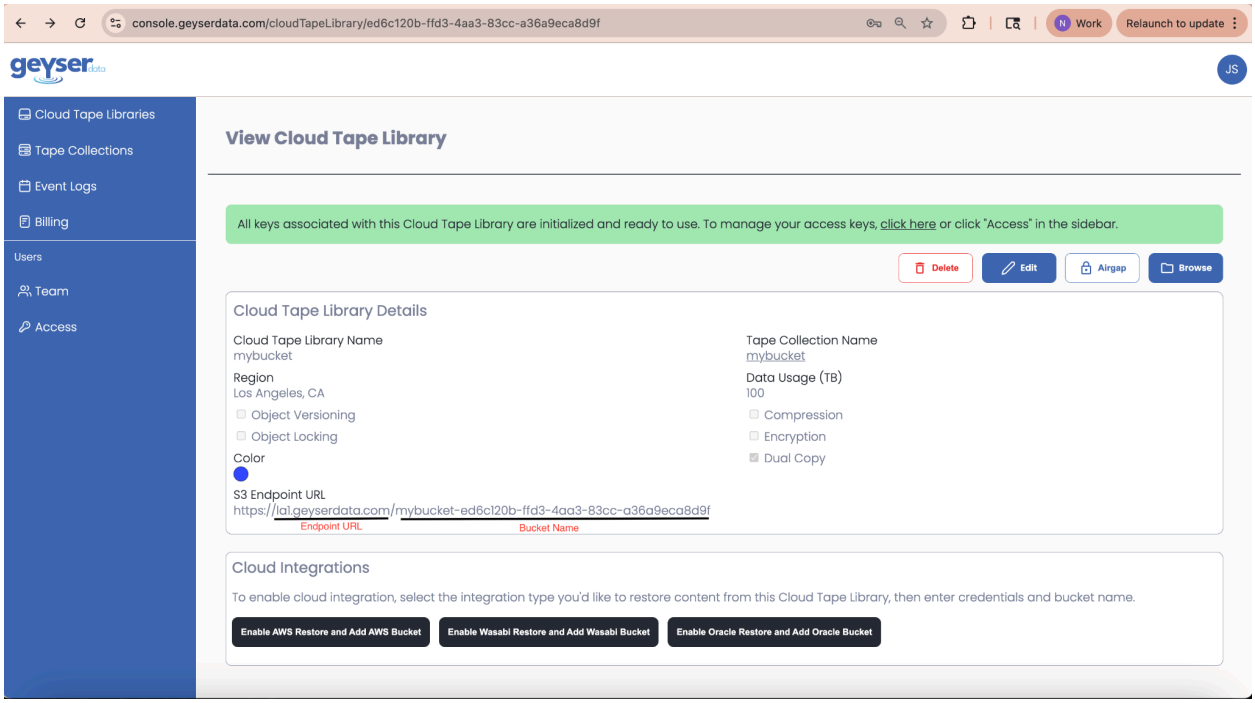
- You are ready to start writing data into your cloud tape library!

Write Data to Your Cloud Tape Library

Geyser Data Cloud Tape Library supports standard S3 Glacier protocols.

You'll need:

- Endpoint URL and Bucket Name



- Access and Secret keys

Sample Python Script to Upload a File

To use this script you need Python version 3 and install the AWS boto3 library with the command:

pip install boto3

To use this script you call it from the cli using:

python upload_file_to_s3.py <filename>

Copy the script below and paste it on a file upload_file_to_s3.py The content of the upload_to_s3.py is below.

Change the value of the following variables to match your own values:

bucket_name, endpoint_url, access_key, secret_key

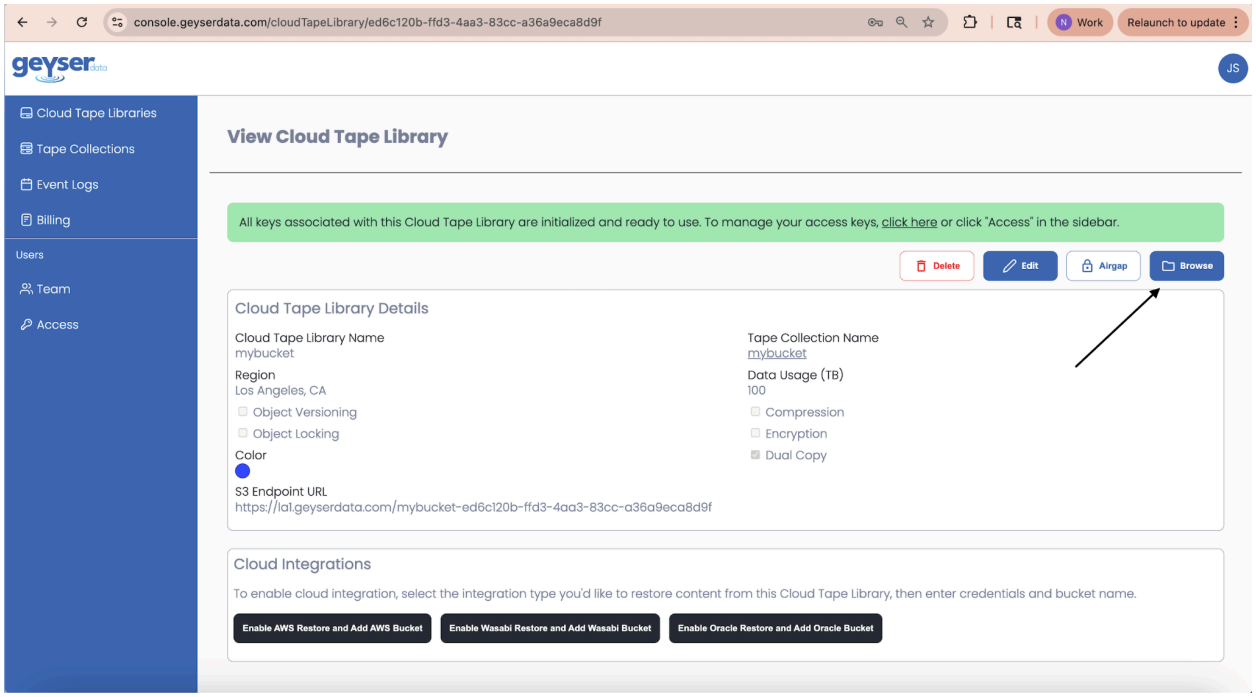
```
import boto3 # type: ignore
import sys
import os

def upload_file_to_s3(filename,bucket_name, endpoint_url, access_key, secret_key):
  # Create a session with the specified parameters
  session = boto3.session.Session()

  # Create an S3 client with custom configurations
  s3_client = session.client(
      service_name='s3',
      aws_access_key_id=access_key,
      aws_secret_access_key=secret_key,
      endpoint_url=endpoint_url,
      verify=False  # Ignore SSL certificates
  )

  # Get the basename of the file to use as the key
  key = os.path.basename(filename)

  try:
      # Upload the file
      s3_client.upload_file(Filename=filename, Bucket=bucket_name, Key=key)
      print(f"File '{filename}' uploaded to bucket '{bucket_name}' successfully.")
  except Exception as e:
      print(f"Error uploading file: {e}")

if __name__ == "__main__":
  if len(sys.argv) != 2:
      print("Usage: python3 upload_file_to_s3.py <filename>")
      sys.exit(1)
   # Variables
  filename = sys.argv[1]
  bucket_name = 'ctluserdemo-b428b54d-fbec-49b1-92ad-ceb520d777cb'
  endpoint_url = 'https://boulderlab.geyserdata.com'
  access_key = 'AKIA47CR2UE7WGLJIRVP'
```

```
secret_key = 'qN8cbYbqf/bUMYIn4iZd3LKFs0Z8hLIrBWGaejW3'


upload_file_to_s3(filename, bucket_name, endpoint_url, access_key, secret_key)
```

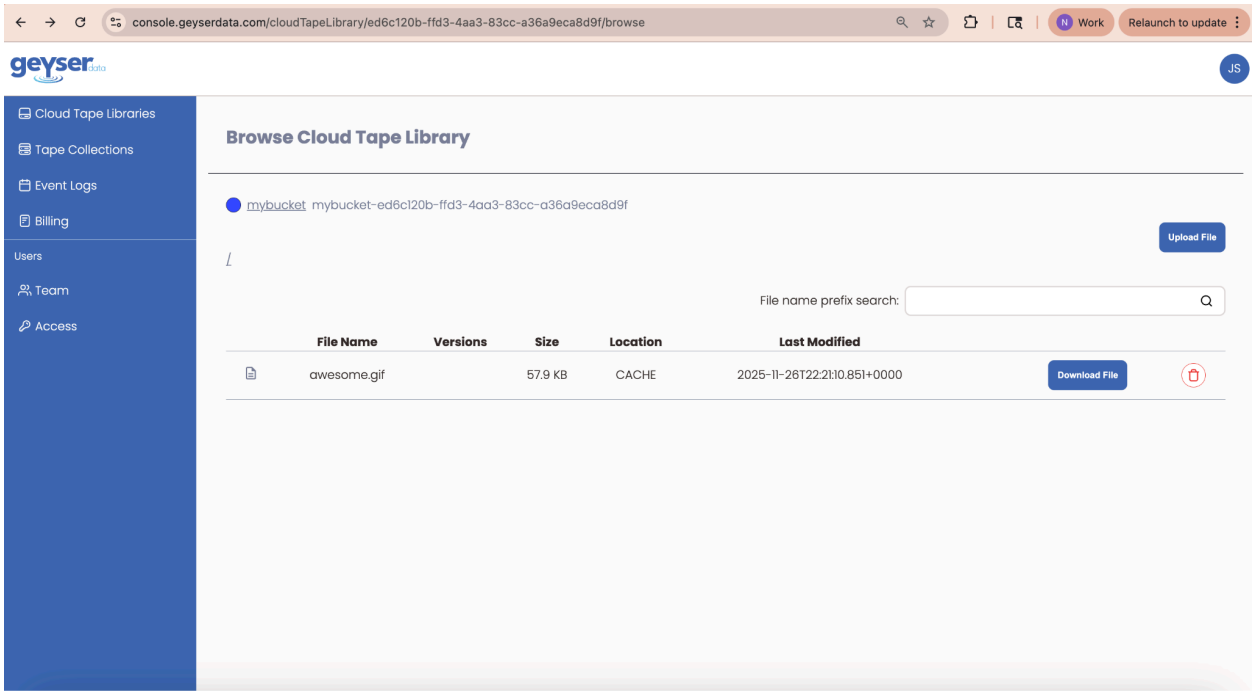
To upload the file test.bin run:
python3 upload_file_to_s3.py test.bin

Browse Data in Your Cloud Tape Library

- Go to the Dashboard, select your library.
- Click on **Browse**



- You will see the files uploaded to your cloud tape library

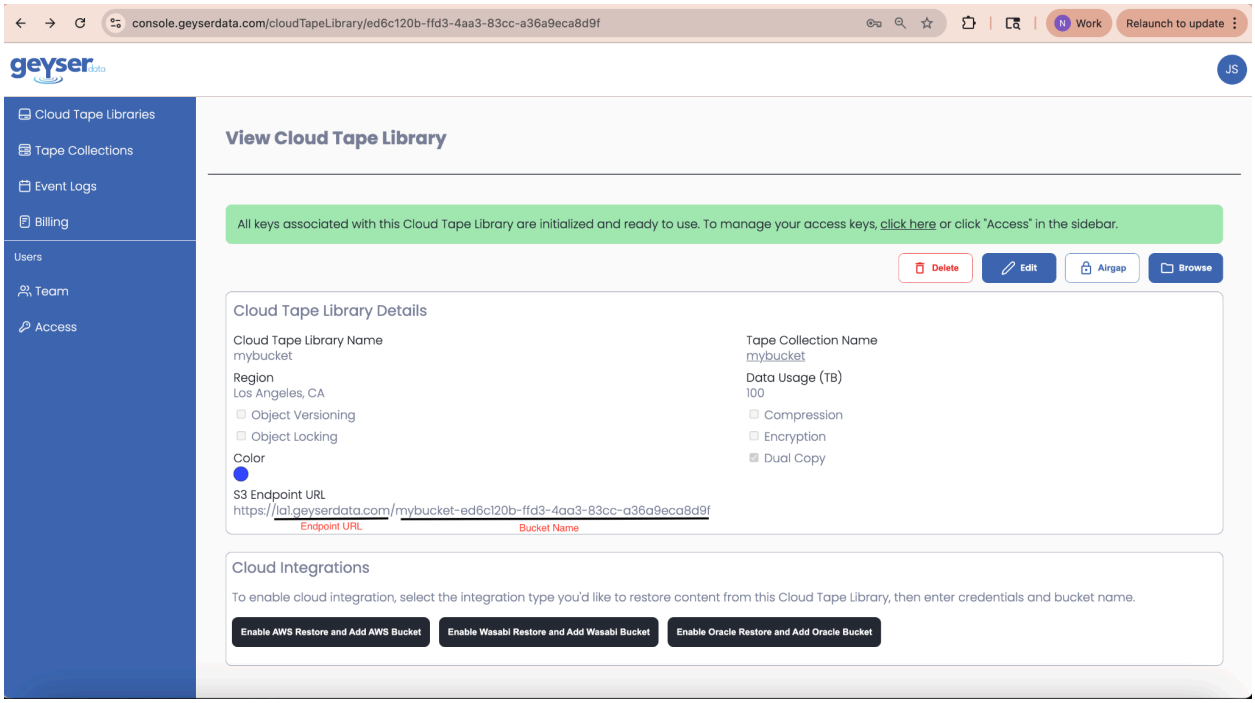


Read Data from Your Cloud Tape Library

Geyser Data Cloud Tape Library supports standard S3 Glacier protocols.

You'll need:

- Endpoint URL and Bucket Name



- Access and Secret keys

Sample Python Script to Upload a File
To use this script you need Python version 3 and install the AWS boto3 library with the command:

pip install boto3

To use this script you can call it with the cli using:

python download_file_from_glacier.py <filename> <local_directory_path>

The code first calls the Glacier Restore API on the particular file, and waits until the file is restored to disk. Once the file is on disk, it immediately calls the GET to download the file.

Copy the script below and paste it on a file download_file_from_glacier.py The content of the download_file_from_glacier.py is below.

Change the value of the following variables to match your own values:
bucket_name, endpoint_url, access_key, secret_key

```
import boto3 # type: ignore
import sys
import os
import time
from botocore.config import Config # type: ignore
from botocore.exceptions import ClientError # type: ignore

def initiate_restore_from_glacier(s3_client, bucket_name, key):
  try:
    # Initiate restore request
    response = s3_client.restore_object(
      Bucket=bucket_name,
      Key=key,
      RestoreRequest={
        'Days': 1,
        'GlacierJobParameters': {
          'Tier': 'Standard'
        }
      }
    )
    print(f"Initiated restore for '{key}' from Glacier.")
  except ClientError as e:
    print(f"Error initiating restore: {e}")
    return False
  return True

def check_restore_status(s3_client, bucket_name, key):
  try:
    response = s3_client.head_object(Bucket=bucket_name, Key=key)
    print(f"\n Response = {response}")
    # Parse the restoration status
    restore_status = response.get('Restore', '')

    if 'ongoing-request="true"' in restore_status:
      print("Restoration is still in progress.")
```

```
            return False
        elif 'ongoing-request="false"' in restore_status:
            # Extract expiry date
            expiry_date = restore_status.split('expiry-date="')[1].split('"')[0]
            print(f"Restoration is complete. Restored object will be available until {expiry_date}.")
            return True
        else:
            print("Restoration information is not available or object is not in Glacier.")
            return True

    except ClientError as e:
        print(f"Error checking restore status: {e}")
        return False

def download_file_from_s3(filename, bucket_name, endpoint_url, access_key, secret_key, download_path):
    # Create a session with the specified parameters
    session = boto3.session.Session()

    # Create a configuration with a long timeout
    config = Config(
        connect_timeout=300,  # 5 minutes
        read_timeout=300      # 5 minutes
    )

    # Create an S3 client with custom configurations
    s3_client = session.client(
        service_name='s3',
        aws_access_key_id=access_key,
        aws_secret_access_key=secret_key,
        endpoint_url=endpoint_url,
        verify=False,  # Ignore SSL certificates
        config=config
    )

    key = filename

    # Initiate the restore process
    if initiate_restore_from_glacier(s3_client, bucket_name, key):
```

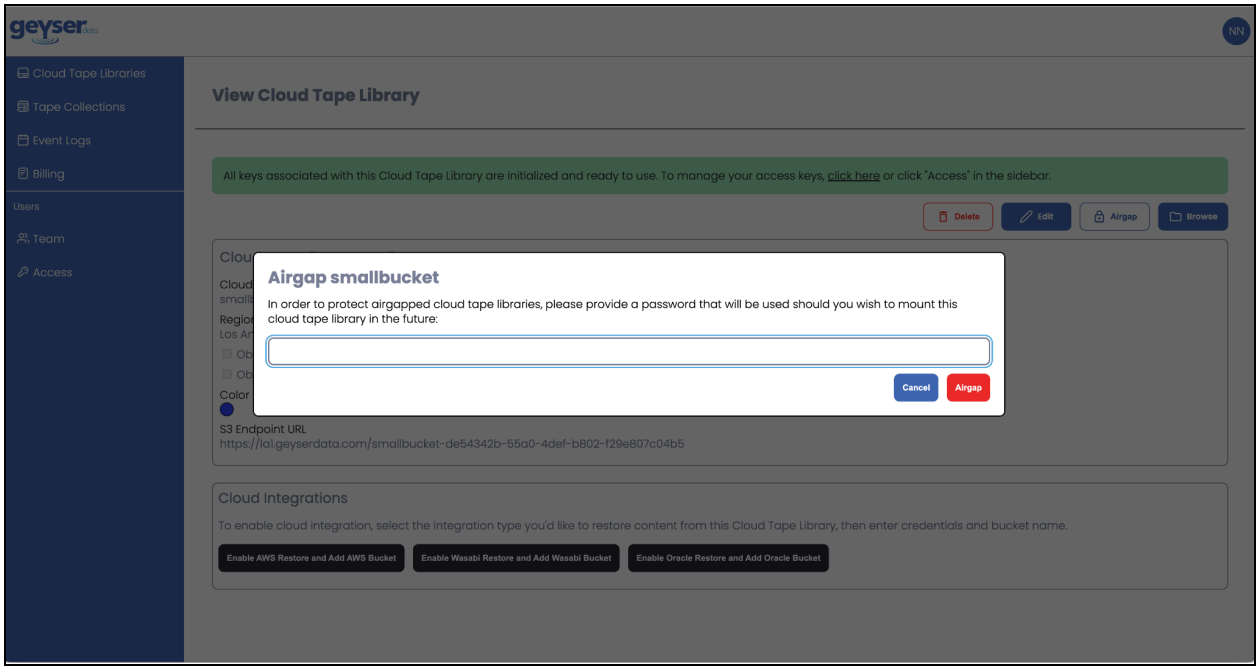
```
    # Check the restore status periodically
    while True:
        if check_restore_status(s3_client, bucket_name, key):
            break
        print("Waiting for 30 seconds before checking the restore status again...")
        time.sleep(30)

try:
    # Download the file
    s3_client.download_file(Bucket=bucket_name, Key=key, Filename=os.path.join(download_path, filename))
    print(f"File '{filename}' downloaded from bucket '{bucket_name}' successfully.")
except Exception as e:
    print(f"Error downloading file: {e}")

if __name__ == "__main__":
    if len(sys.argv) != 3:
        print("Usage: python download_file_from_glacier.py <filename> <download_path>")
        sys.exit(1)

    filename = sys.argv[1]
    download_path = sys.argv[2]
    bucket_name = 'latestsinglecopy-774c5929-a972-4e59-b300-de4087b6b664'
    endpoint_url = 'https://la1.geyserdata.com'
    access_key = 'AKZA27CR2U87Q4Z26MVG'
    secret_key = 'r7glObzOSgkdfa9bUA0KZFk7+wbdNG/YkGX1gscy'

    start_time = time.time()

    download_file_from_s3(filename, bucket_name, endpoint_url, access_key, secret_key, download_path)
    end_time = time.time()

    elapsed_time = end_time - start_time
    print(f"Total execution time: {elapsed_time:.4f} seconds")
```

To restore and download a file:
python3 download_file_from_glacier.py test.bin ./

Air Gap Feature

Geyser Data enables you to Air Gap a Cloud Tape Library for enhanced security. When a Cloud Tape Library is Air gapped, its tape cartridges are unmounted from the virtual tape drives, preventing any reads or writes and providing strong protection against unauthorized access or cyber attacks.

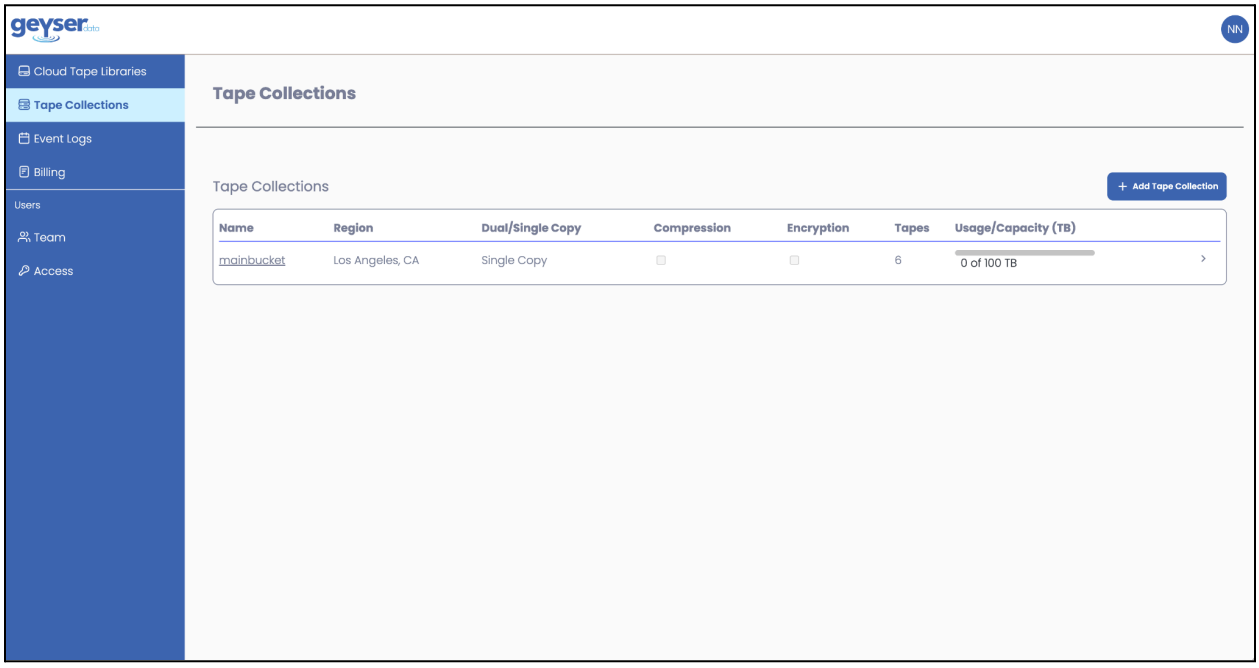
To Air Gap a cloud tape library, simply click the **Air Gap** button and enter a unique password for restoration. Be sure to record this password in a secure location, as you will need it to restore access; without it, the library cannot be remounted.



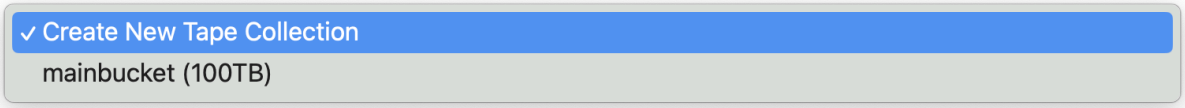
To restore access, repeat the process: select the Cloud Tape Library, click **Air Gap** and enter the same air gap password to re-enable read and write capabilities.

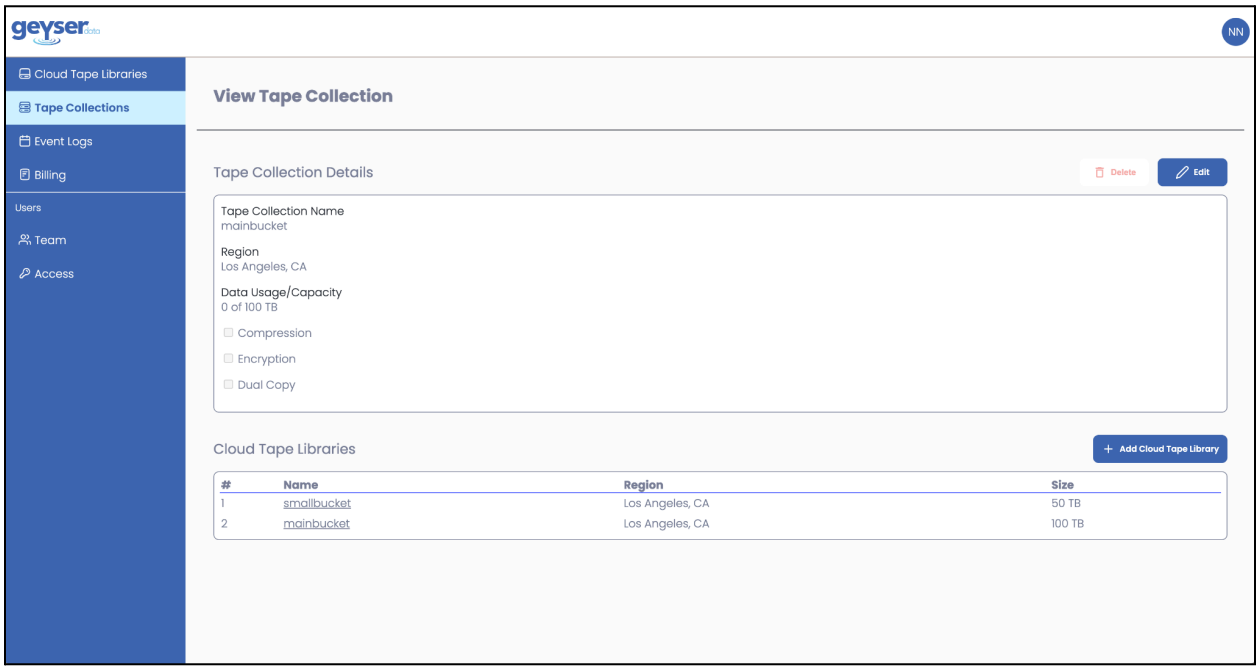
Tape Collections

A Tape collection is a set of Physical Tape Cartridges mapped to one or more Cloud Tape Libraries (i.e. buckets). Creating multiple Tape Collections helps separate workloads so that data for each is stored contiguously on tape, enabling more efficient data retrieval. For optimal performance, assign a dedicated Tape Collection to each Cloud Tape Library. However, for smaller buckets, you can make several Cloud Tape Libraries onto the same Tape Collection.



When you create your first Tape Collection along with the first Cloud Tape Library, you can later add additional libraries to that collection. In the Add Cloud Tape Library screen, simply select the desired Tape Collection.





For example, a single Tape Collection named "mainbucket" may have 2 Cloud Tape Libraries mapped to it: "main bucket" and "small bucket."

Useful links

[How to use Cyberduck with Geyser data]

[S3 Browser Setup Guide]

[Veeam Integration Guide]

[Cohesity Integration Guide]

[MinIO Integration Guide]

Support

Have questions or need help?
Contact: support@geyserdata.com