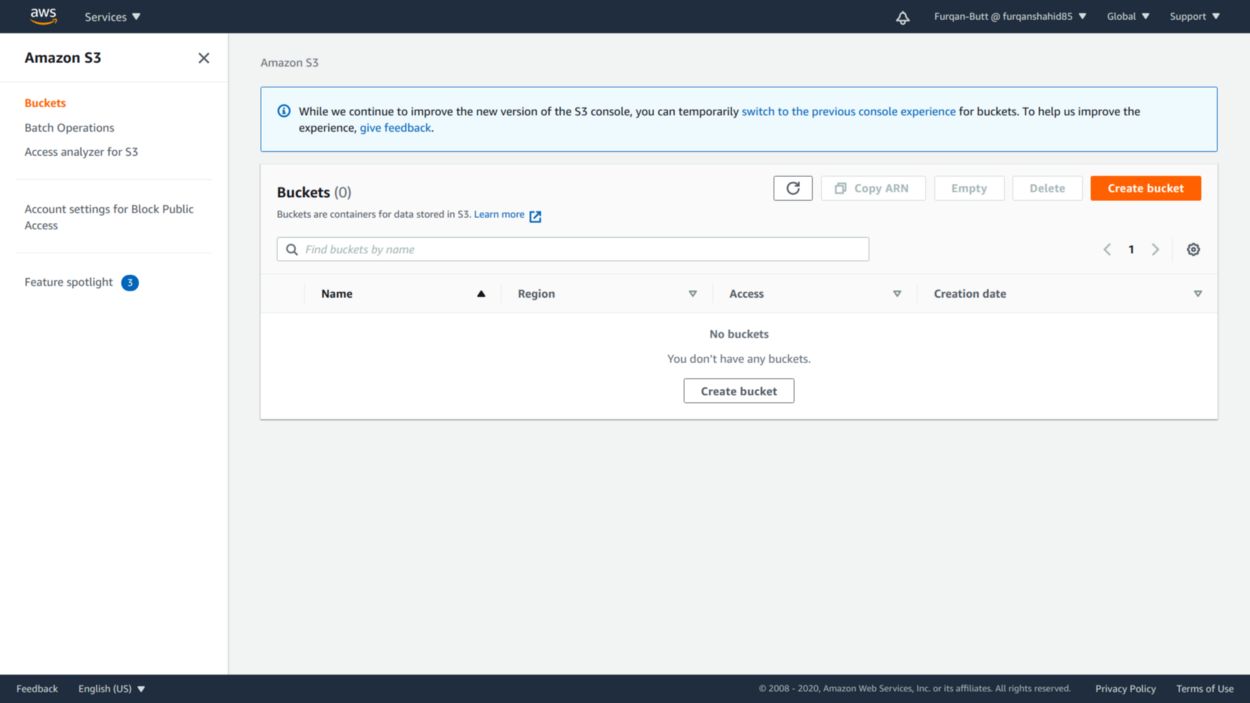
Creating Buckets and Uploading data to S3

S3 bucket Creation

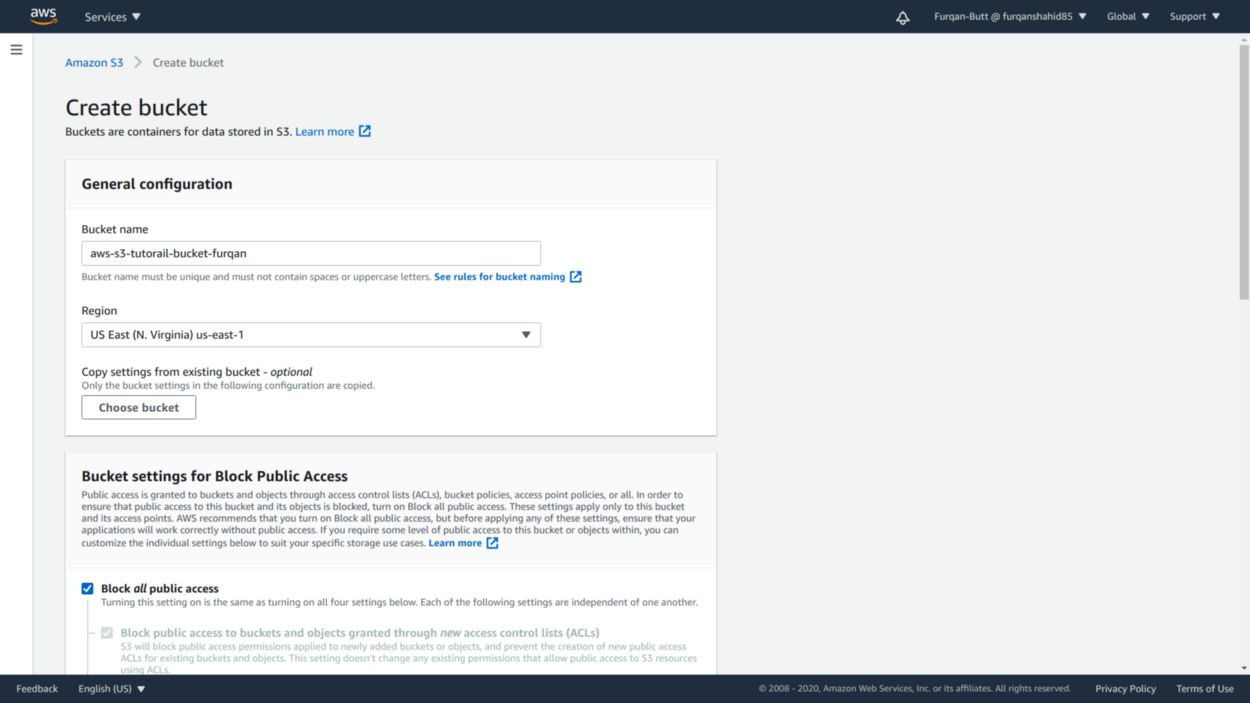
S3 is one of the most user-friendly service in the AWS ecosystem. We have multiple options available to upload data to S3 which include manually uploading data using the **Management Console**or uploading programmatically via S3 APIs, SDKs, and AWS CLI.

* After you log in to your AWS account search for the S3 service from the services dropdown at the top you’ll land on the following page:



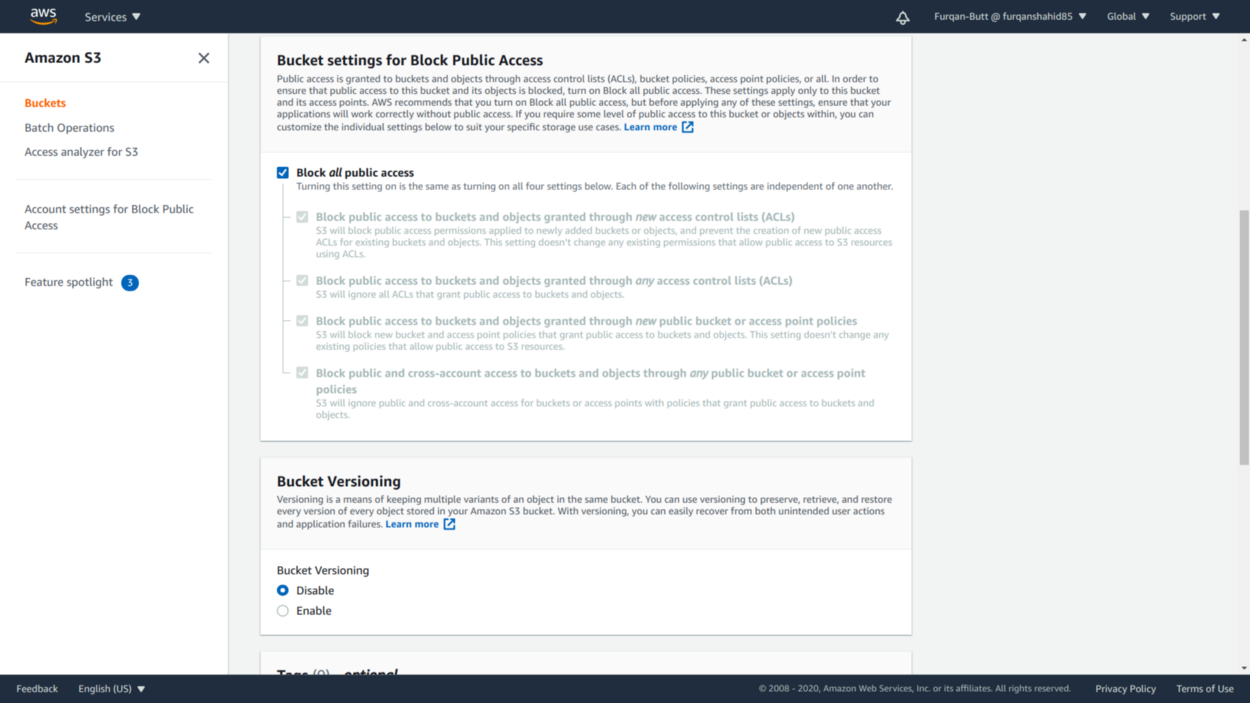
S3 home screen

* As you can see there are zero buckets so we’ll create a new one. Click on the **Create bucket** button at the bottom



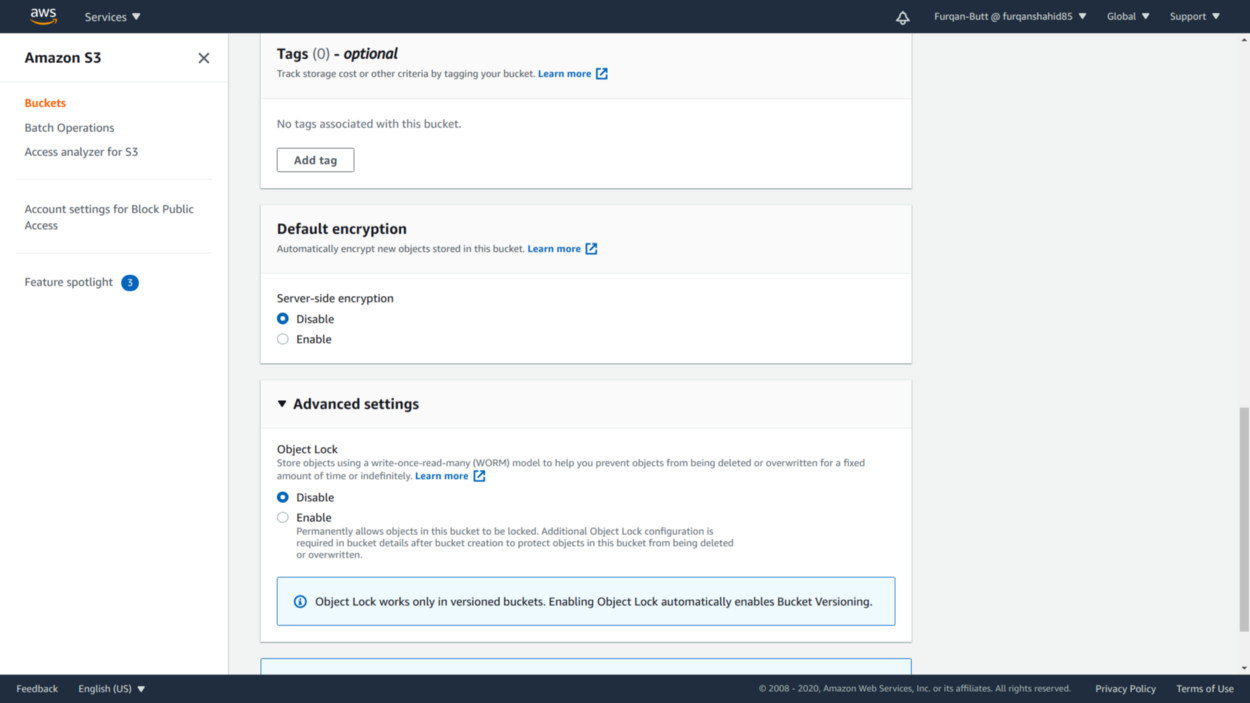
Creating an S3 bucket

* From the current screen provide the **bucket name** ([naming conventions for bucket names](https://docs.aws.amazon.com/AmazonS3/latest/dev/BucketRestrictions.html)) and select the **region** to create the bucket. One thing to note here is that S3 is a **global service**as seen at the top right but buckets are created in a specific region. This helps reduce **latency**. Bucket names must be **unique globally**and no two buckets in AWS S3 have the same name. Some other configurations we can set here are as follows:

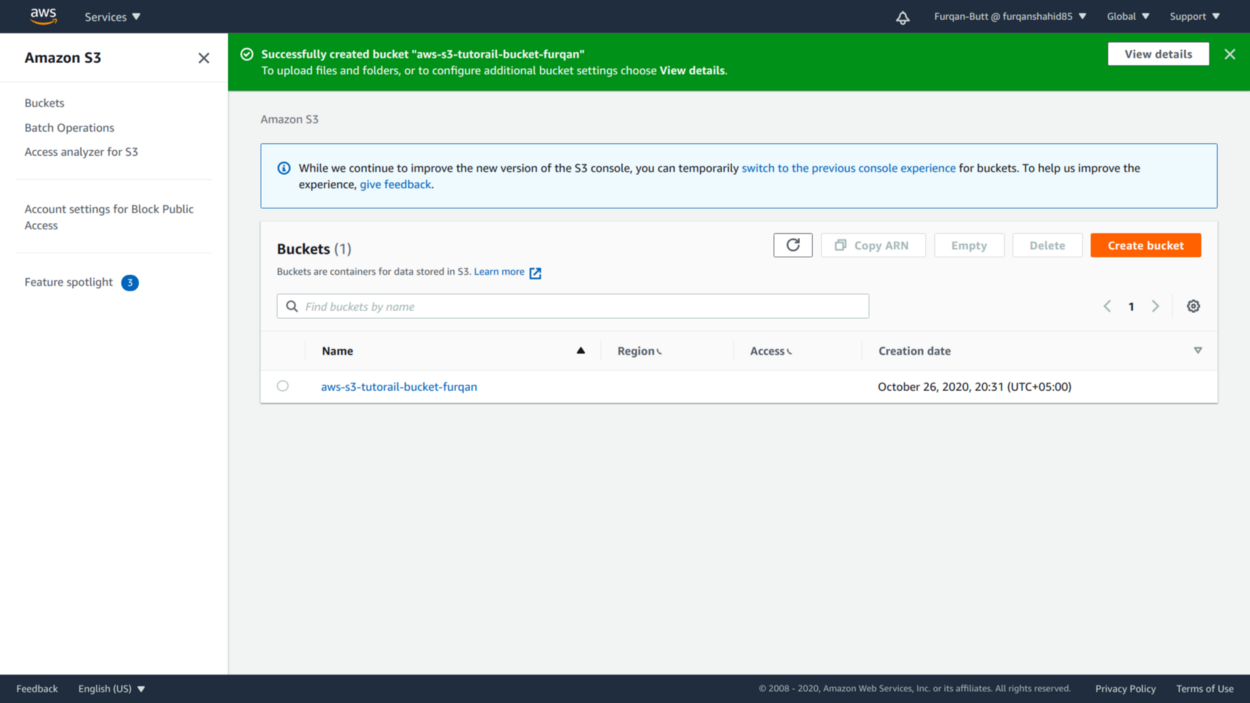


Blocking Public Access (Private By Default) and Setting Bucket Versioning

* Bucket access must be carefully set and we can also enable bucket versioning at bucket creation time (can be changed after creation).



* We can optionally set tags(up to 50) for better management of buckets and also enable object encryption (can be set after creation). Object locking can also be enabled at creation time. After setting all configurations, click on the **Create bucket** button at the bottom to create the bucket.



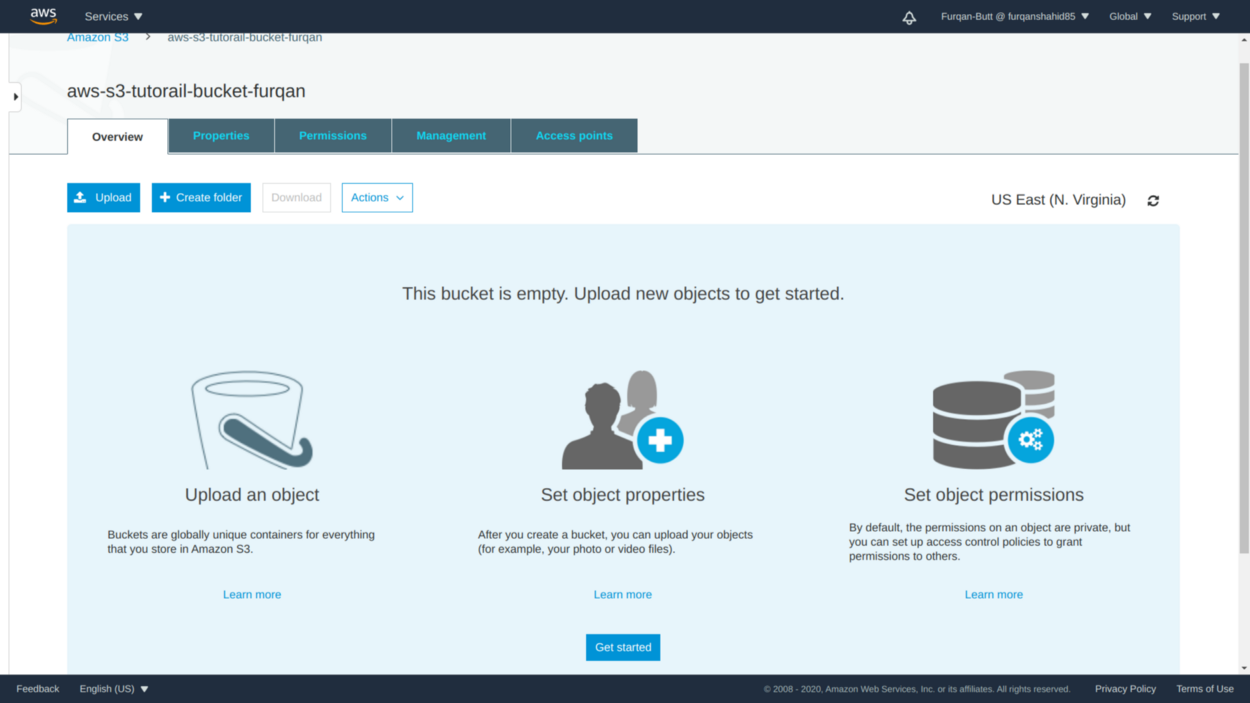
Bucket creation success.

As we can see bucket is created successfully. Now, we’ll proceed to upload data to our newly created bucket.

Uploading data to S3

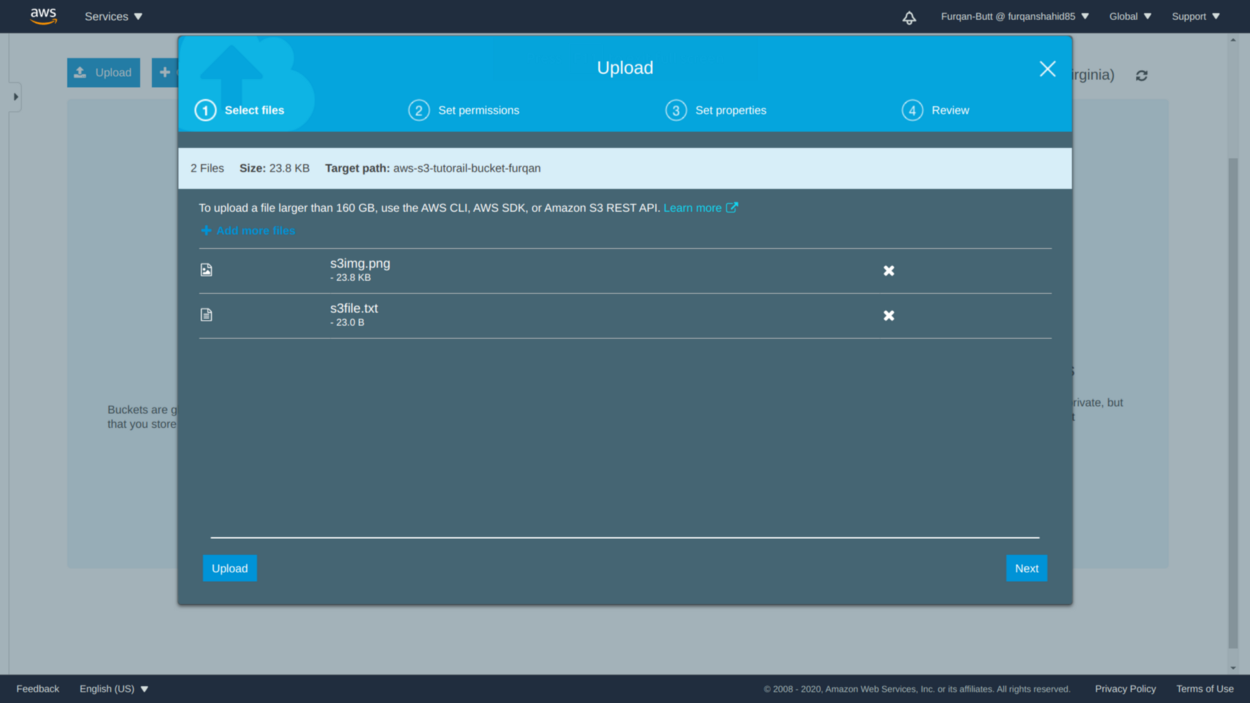
Now that we have created a bucket we can upload some sample data on s3 which is fairly simple…

* Click on your created bucket name which will take you to the screen below



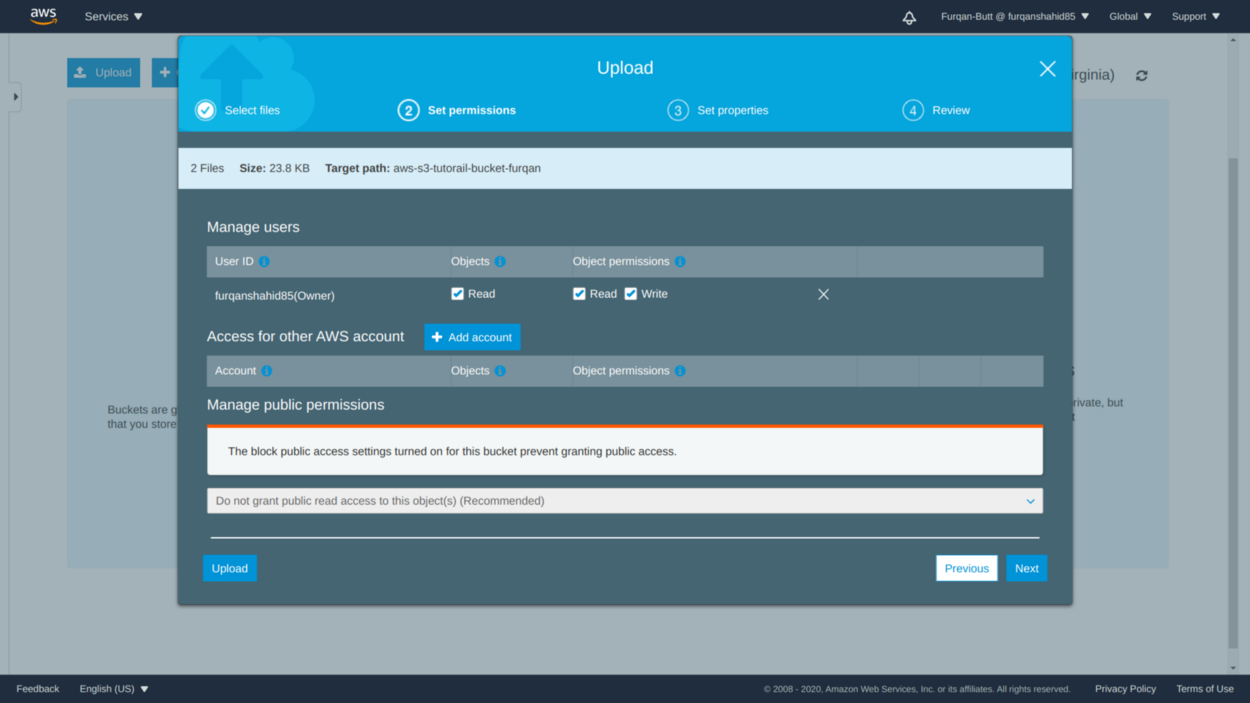
S3 image home

* We can either upload files using the upload button or even drag and drop into it. Other methods are uploading using AWS CLI, SDKs, or S3 REST APIs which we’re not covering here.
* I’m uploading 2 files here an image file and a text file. We can see S3 either lets you upload directly or set some optional settings before uploading. **Files larger than 160GB can’t be uploaded via Management Console.**

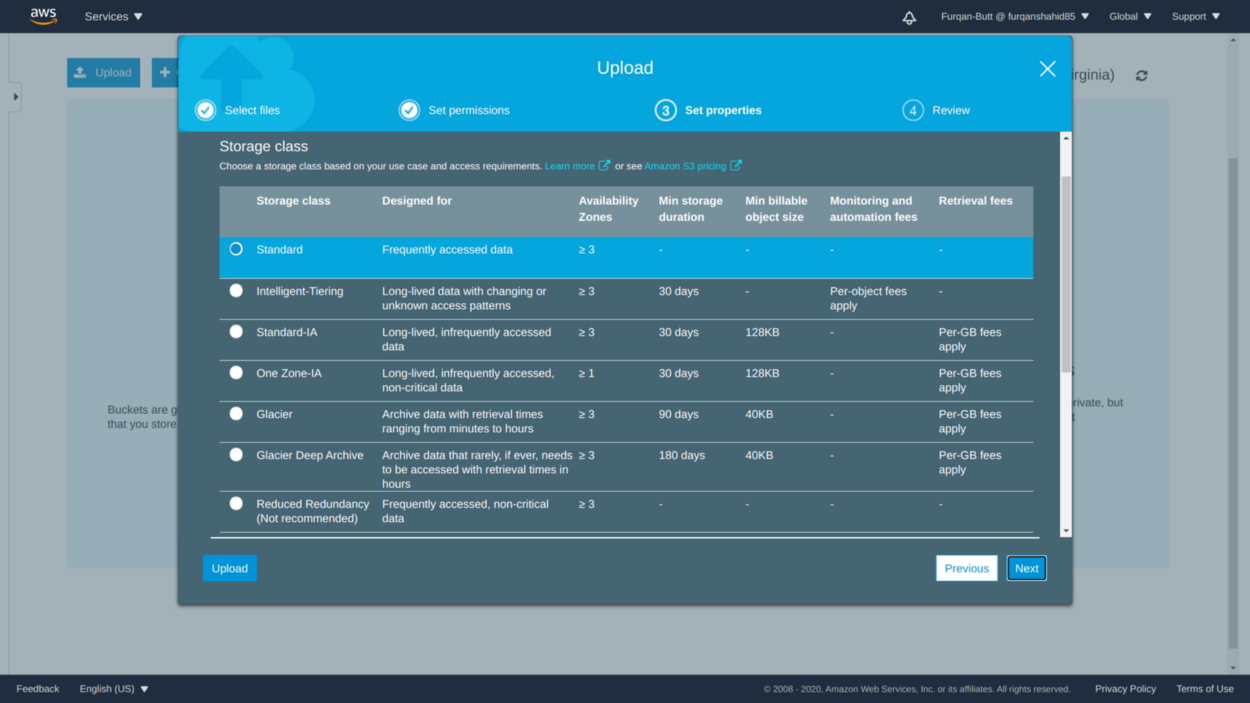


Data upload on S3

* Here we can set user-level access and set which users to allow access and what kind of access to these objects. By default, all public access is blocked.

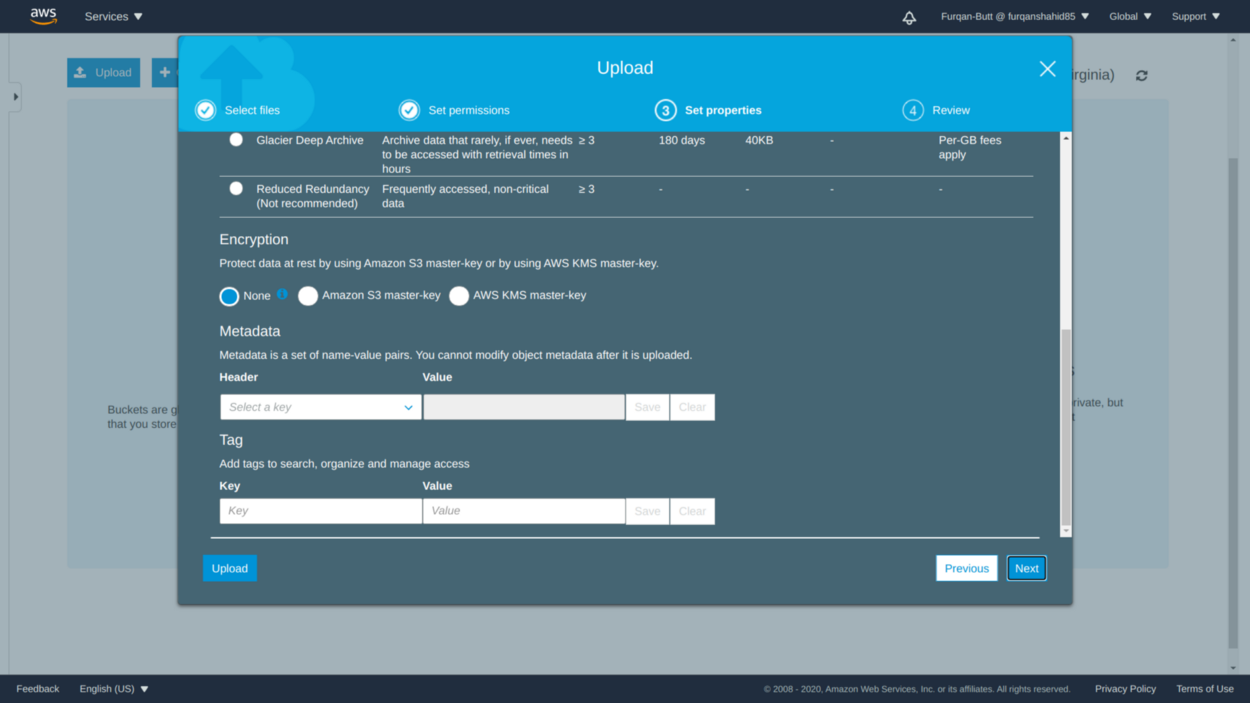


* Here we can set the storage classes for these objects. By default, the objects are assigned to “standard storage class”.



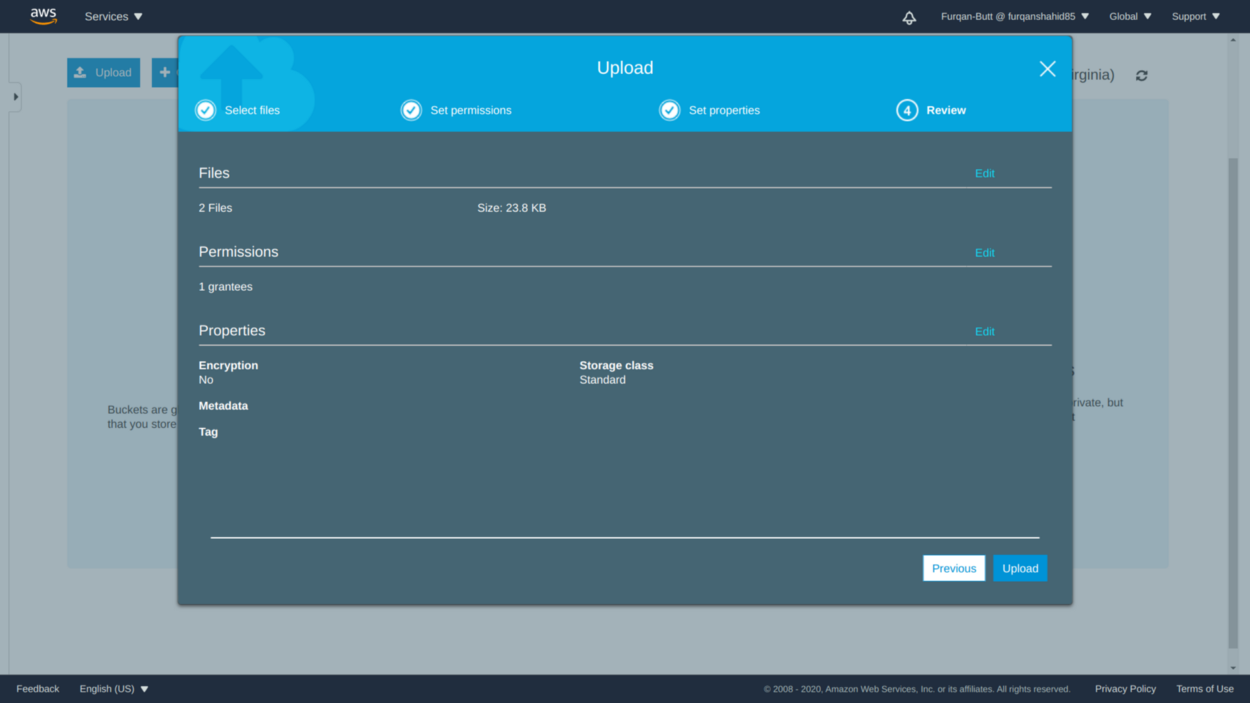
Setting up S3 objects storage classes

* Here can enable encryption for these objects. The objects will be encrypted before being saved. We can also set meta-data and tags for these objects.



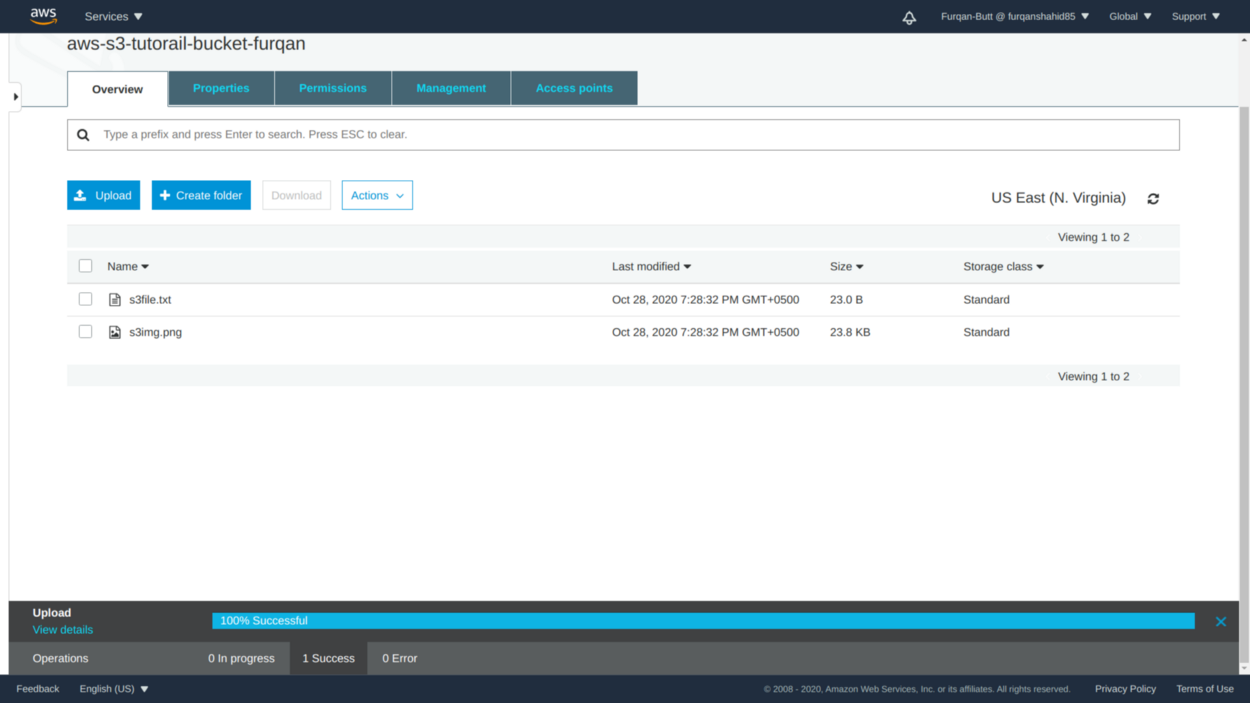
Setting up encryption, meta-data, tags.

* The summary for all setting setup for these files.



Objects settings summary.

* As we can see files have been uploaded successfully.



Objects Upload success.

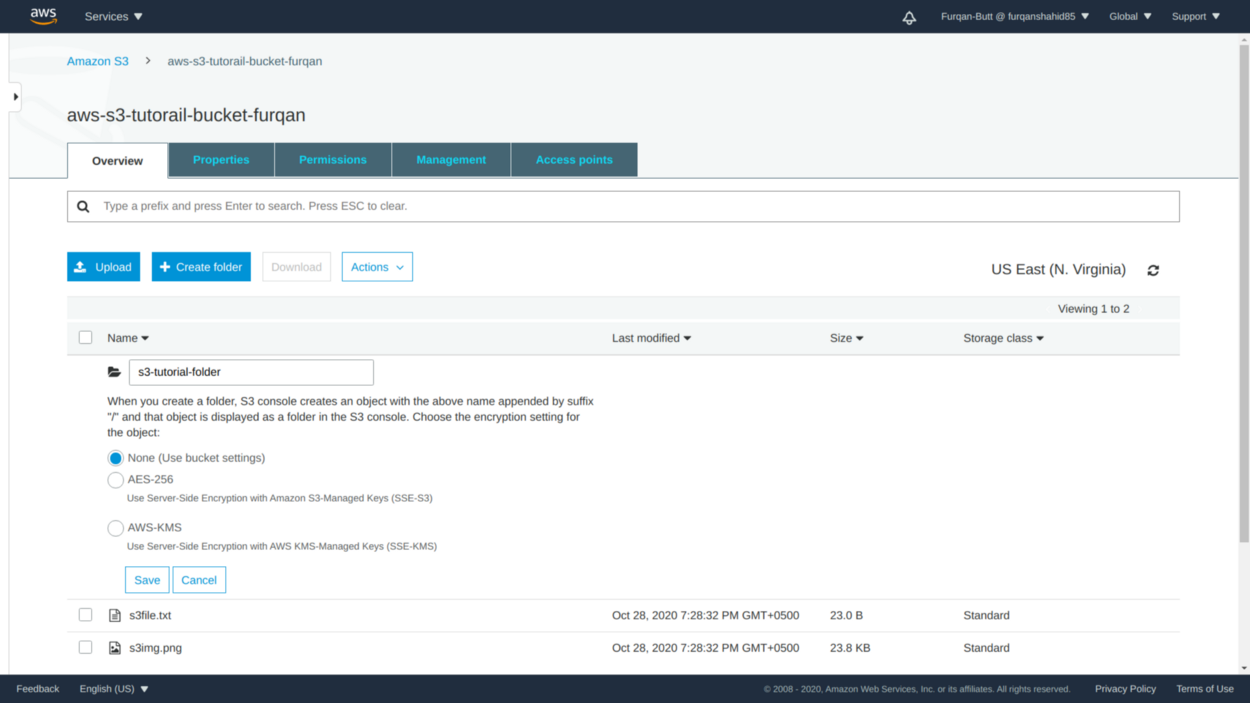
Buckets and Object Keys Structuring

S3 allows us to set up a directory structure for better management of data although physically, the files are not stored in directories.

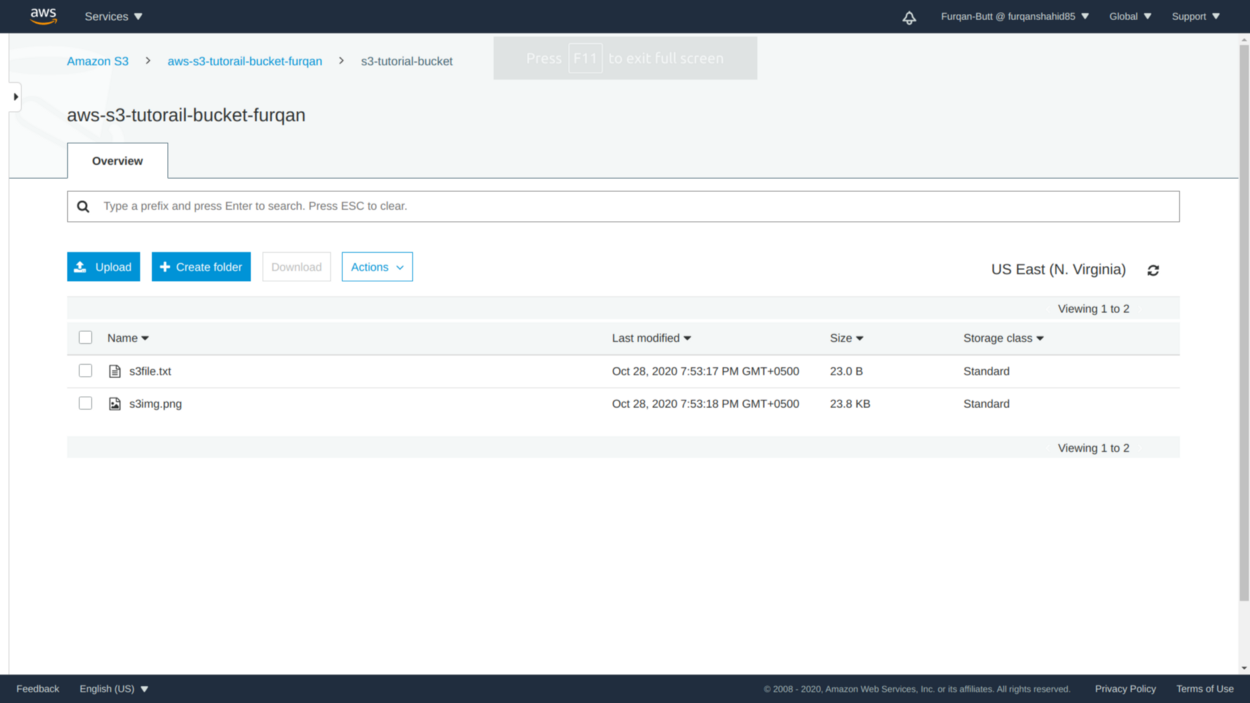
* Creating a folder is simple enough. Click on the **create folder** button. We set up the folder name and the folder is created. All objects with the folder will have a key like:

**s3://aws-s3-tutorail-bucket-furqan/s3-tutorial-folder/s3file.txt**

* We can also set encryption at the folder level.

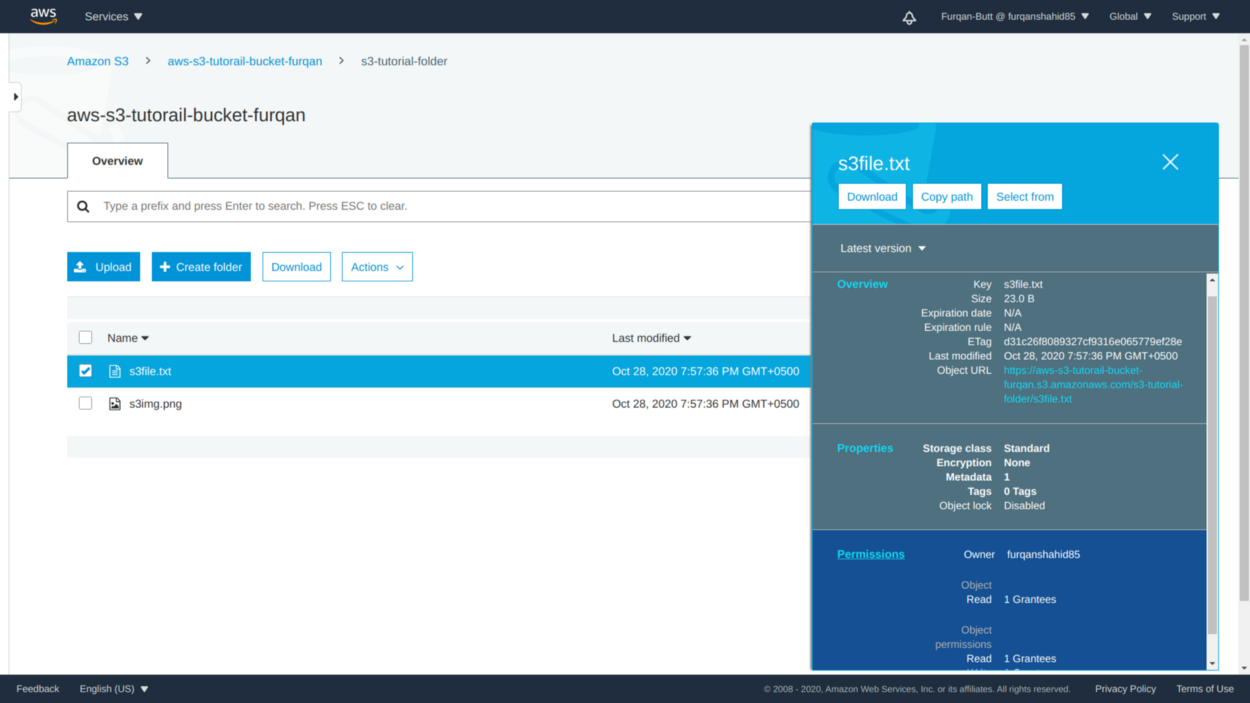


Creating an S3 Folder



Files moved to the folder

* Selecting an object displays object relevant details.



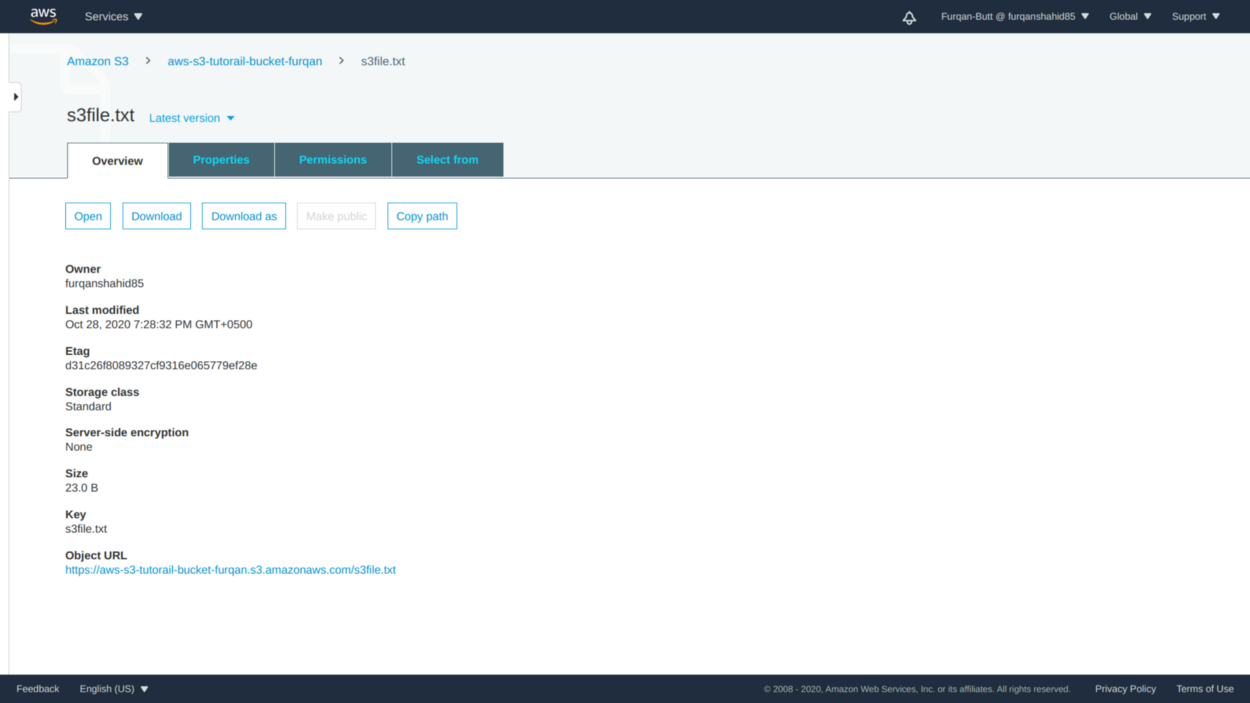
Object Overview.

Exploring S3 Storage Classes and Life Cycle Management

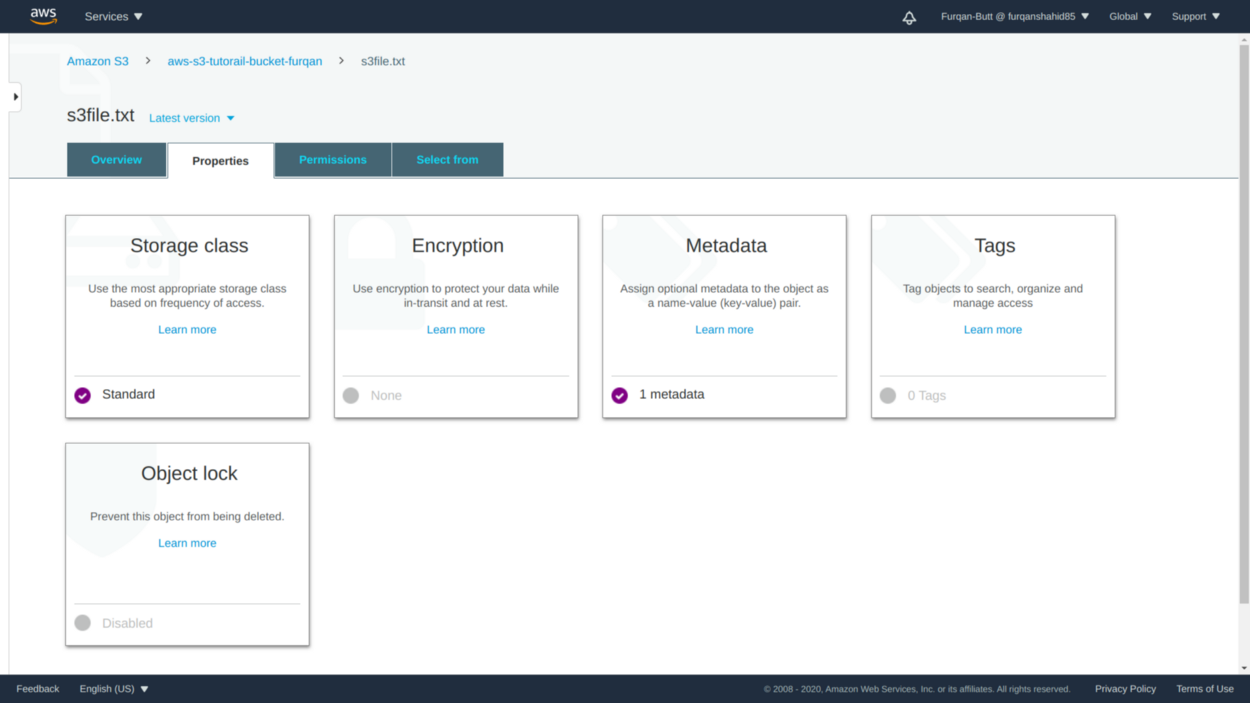
S3 offers a range of storage classes that are purposely built for optimizing the cost of storage. All objects in S3 might not be accessed very frequently hence setting up appropriate storage can help reduce the cost. Let’s explore how we can edit the storage class of an object:

\*\**AWS Changed their S3 Console design while I was midway writing this article anyhow -\_\_-*

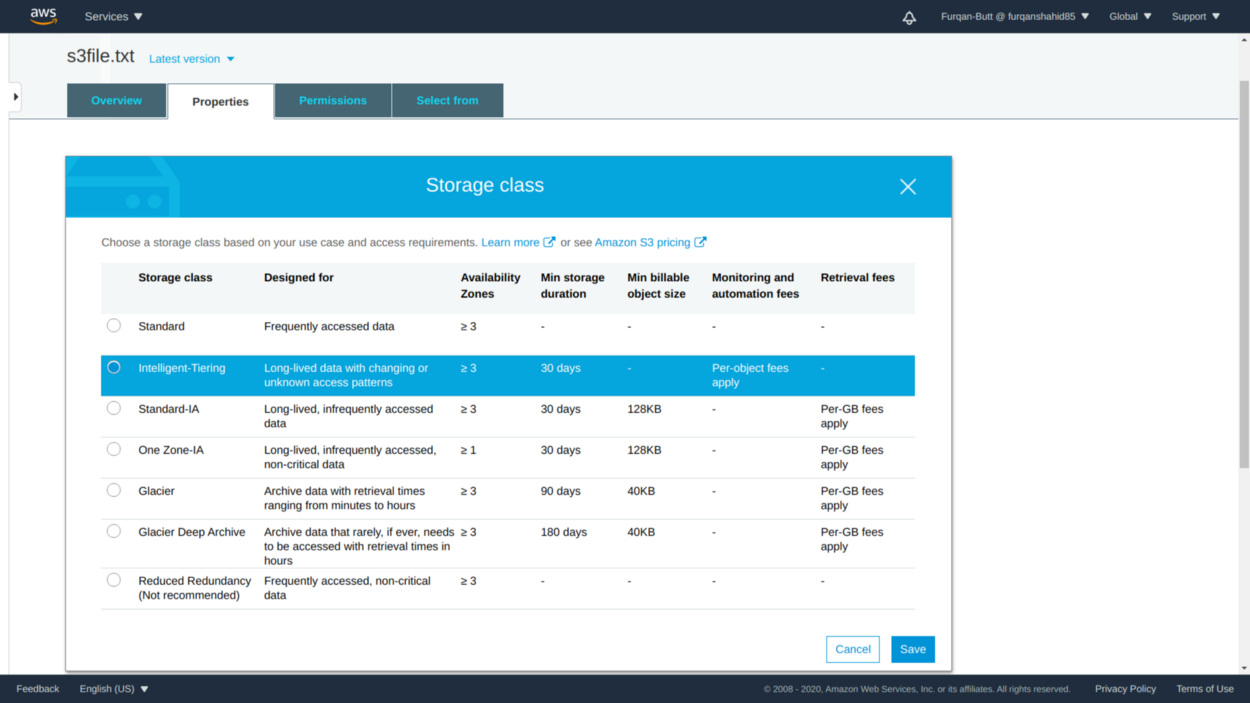
* Select the object whose storage class is to be changed. There are 7 different storage classes that can be selected. Just for the sake of this tutorial I have selected the **Intelligent Tiering** storage class
* This is an object-level operation and we can change the storage class of one or more objects at the same time or even all objects under a folder.
* Multiple objects can have different storage classes at the same time in the same bucket.



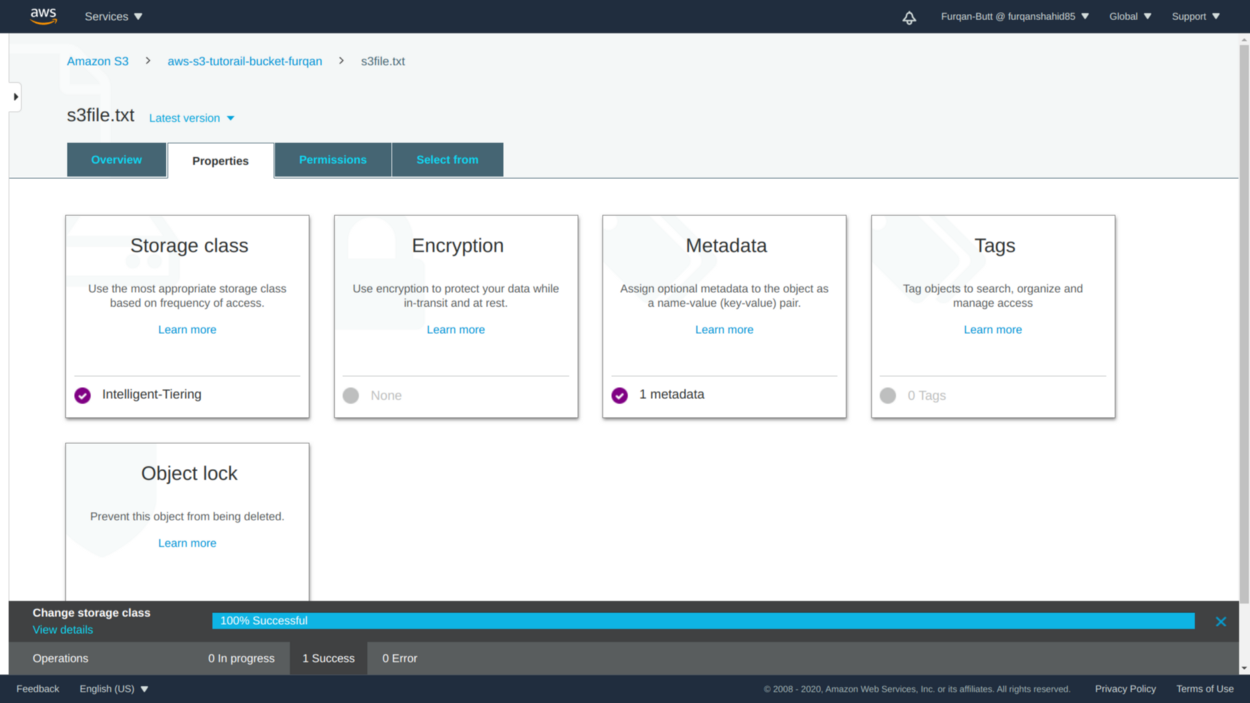
1. Changing Storage Class



2.



3. Changing storage class from standard to intelligent tiering

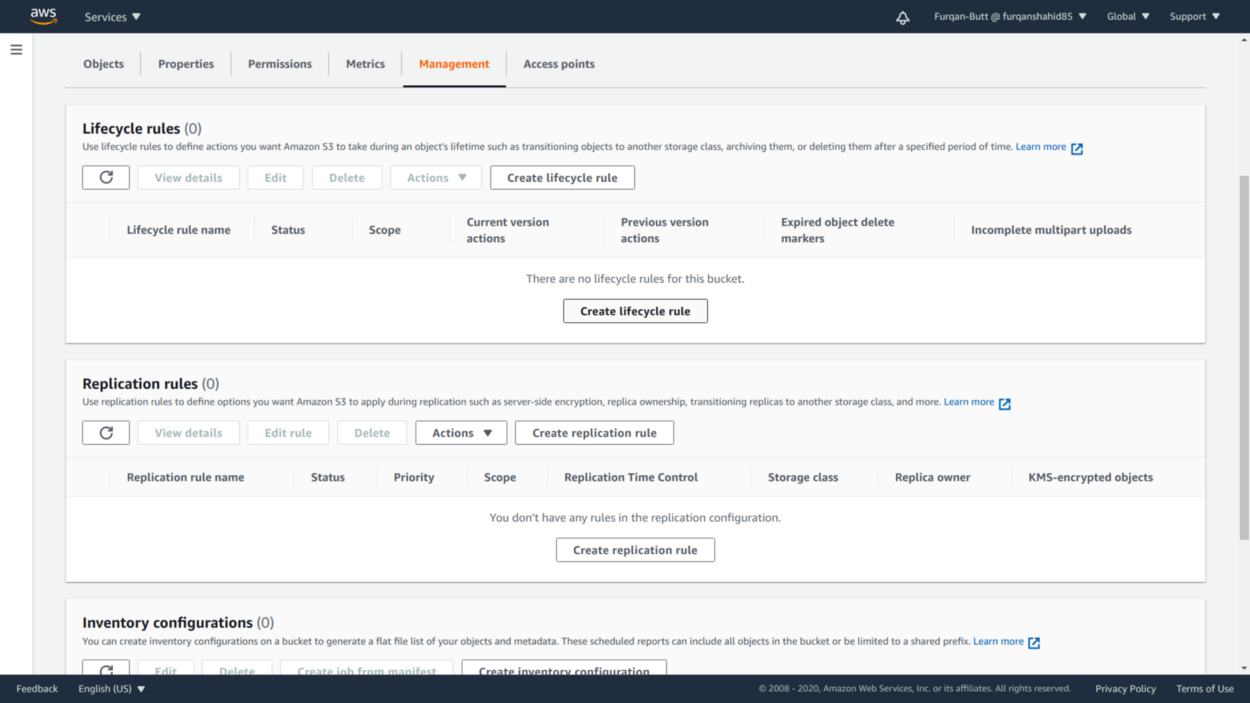


4. Storage Class Changed to Intelligent Tiering.

Setting up Life Cycle Management Configuration

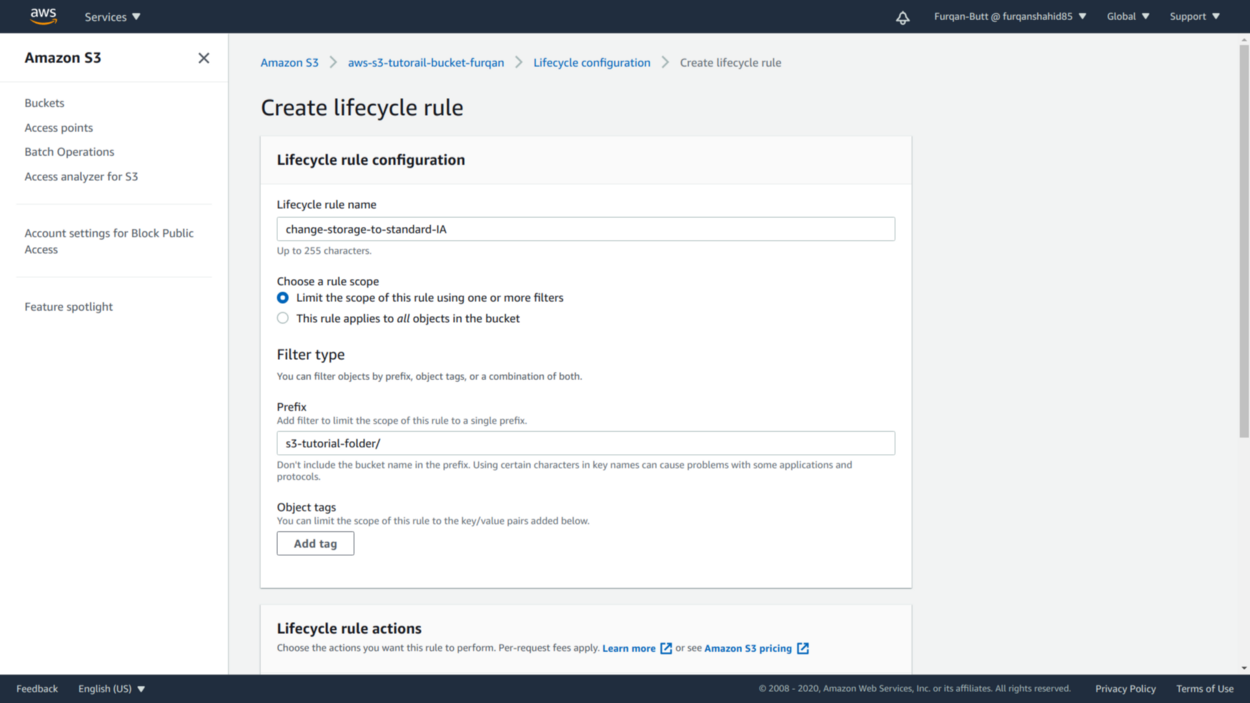
Instead of manually setting up the storage classes for each individual objects, S3 also provides an option to set up **Life Cycle Rules**that will automatically change the storage class of objects under a bucket after a specific interval of time.

* Goto the management tab in your S3 bucket and under the **life cycle rules**click the **create life cycle rule**button

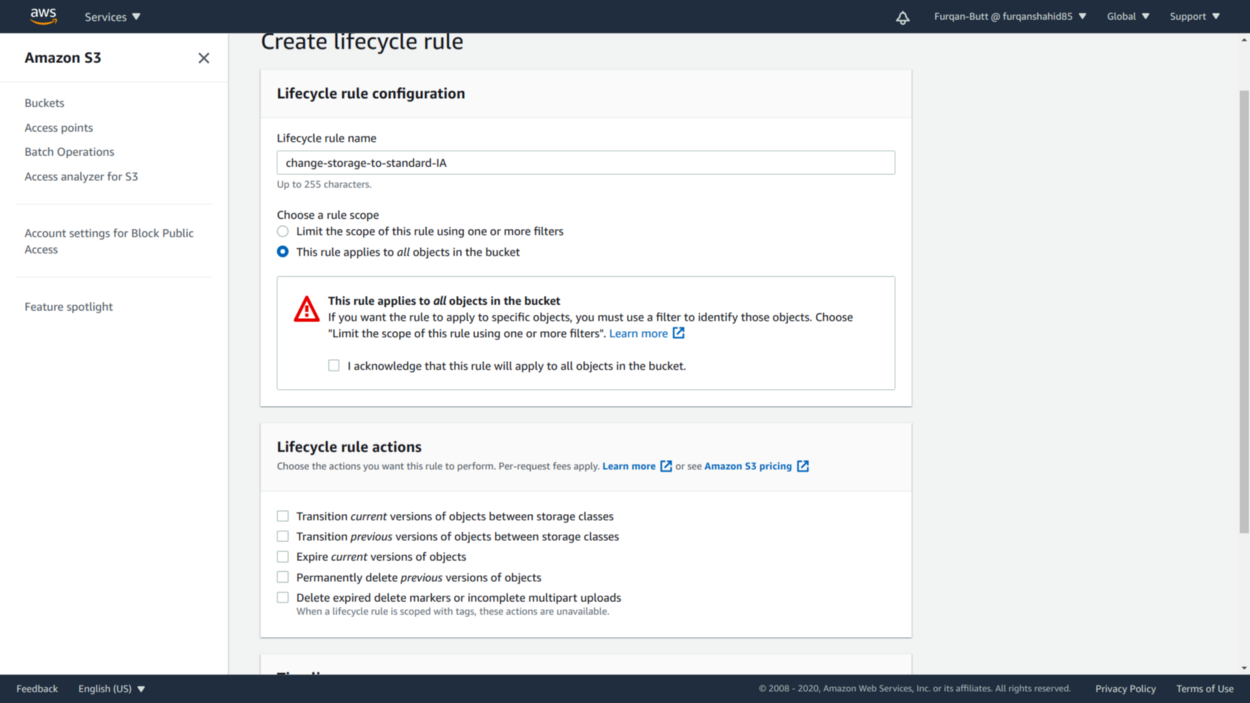


1. Setting up life cycle rules.

* Provide a name for the rule. The life cycle rules can be applied to specific objects by setting up the filter condition or we can apply them to all objects in the bucket.

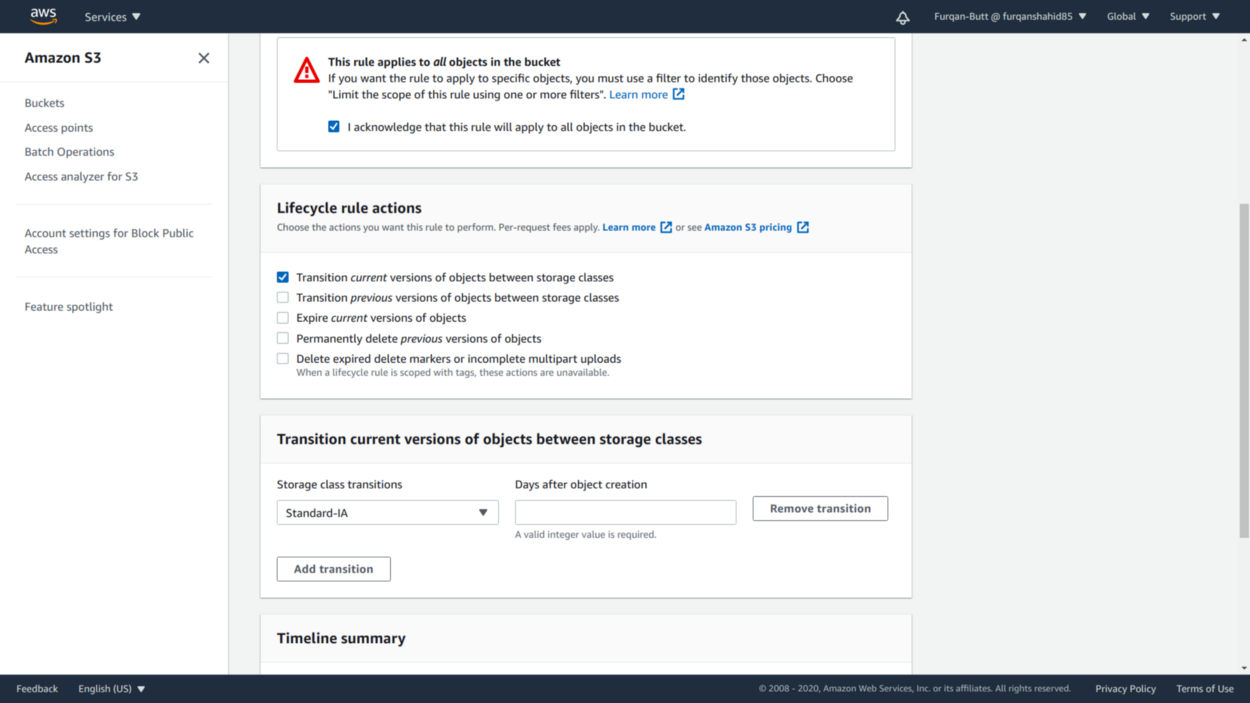


2.a Filter Specific Objects to apply rule.

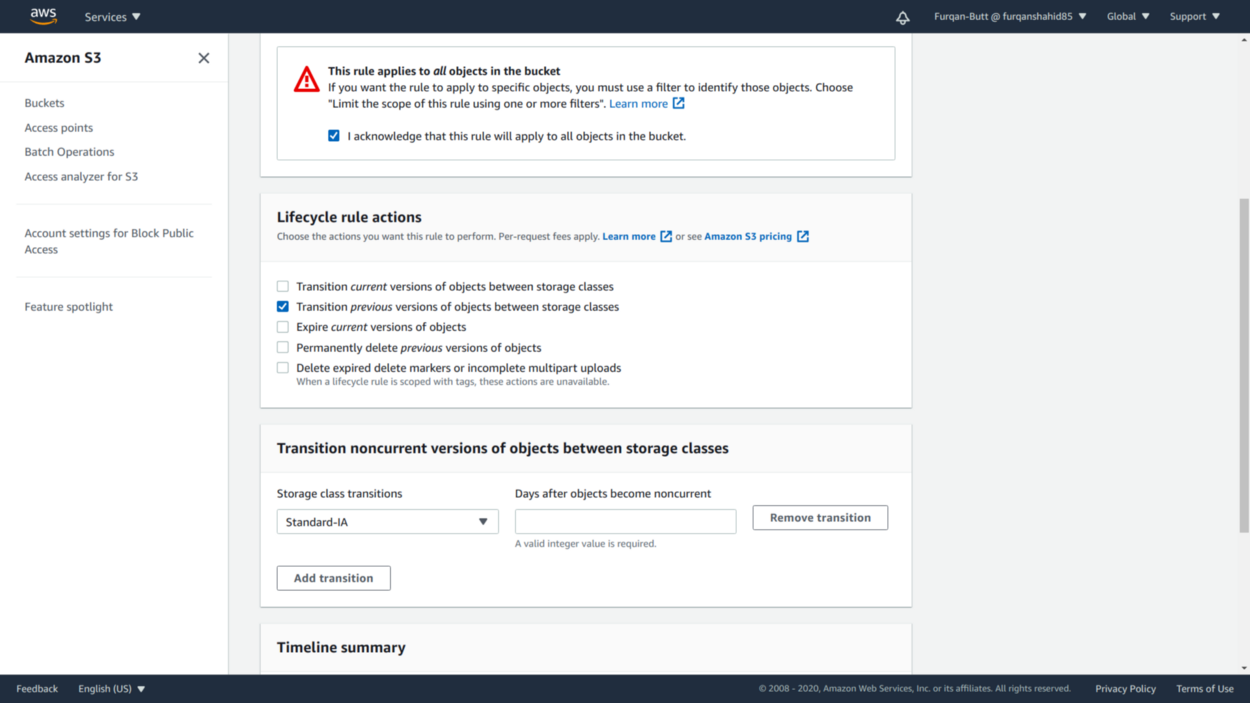


2.b Rule applied to all the objects.

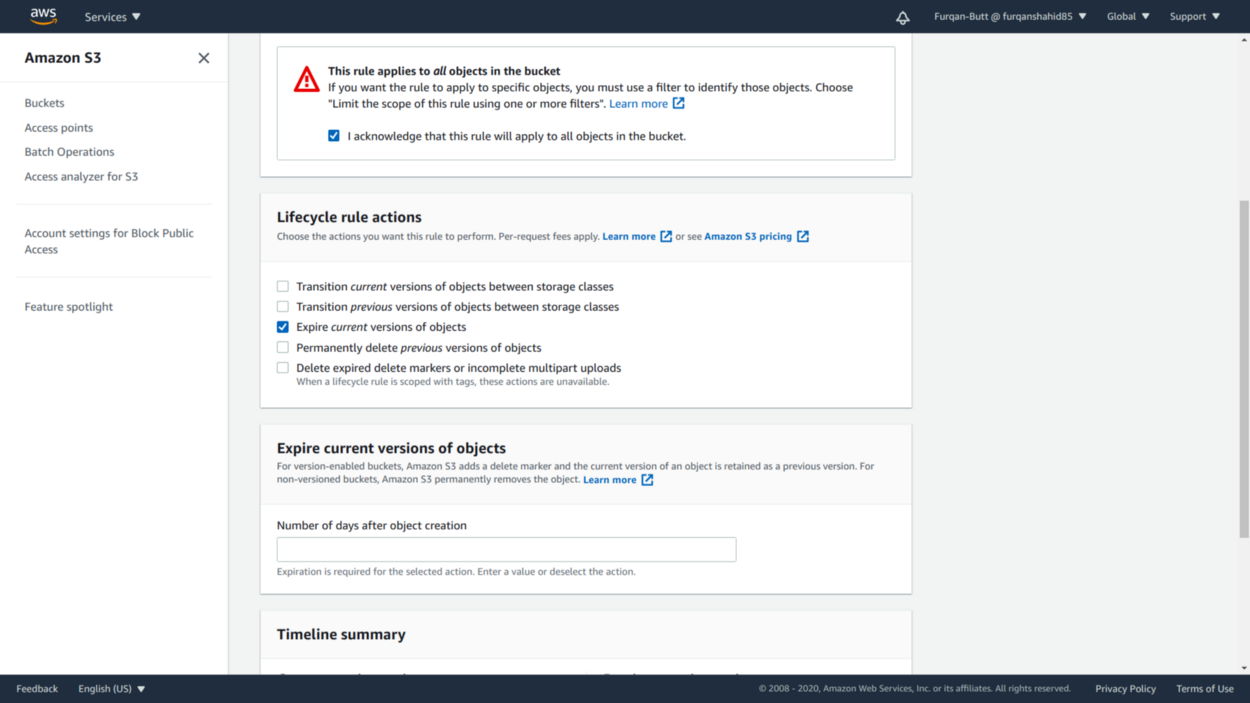
* After selecting what objects to apply the rule to we select what rules to apply.
* We have transitions as well as expiration rules for objects. Transition rules will change the current storage class to the specified class after the specified number of days while expiration rules will delete or expire the objects after the specified number of days.



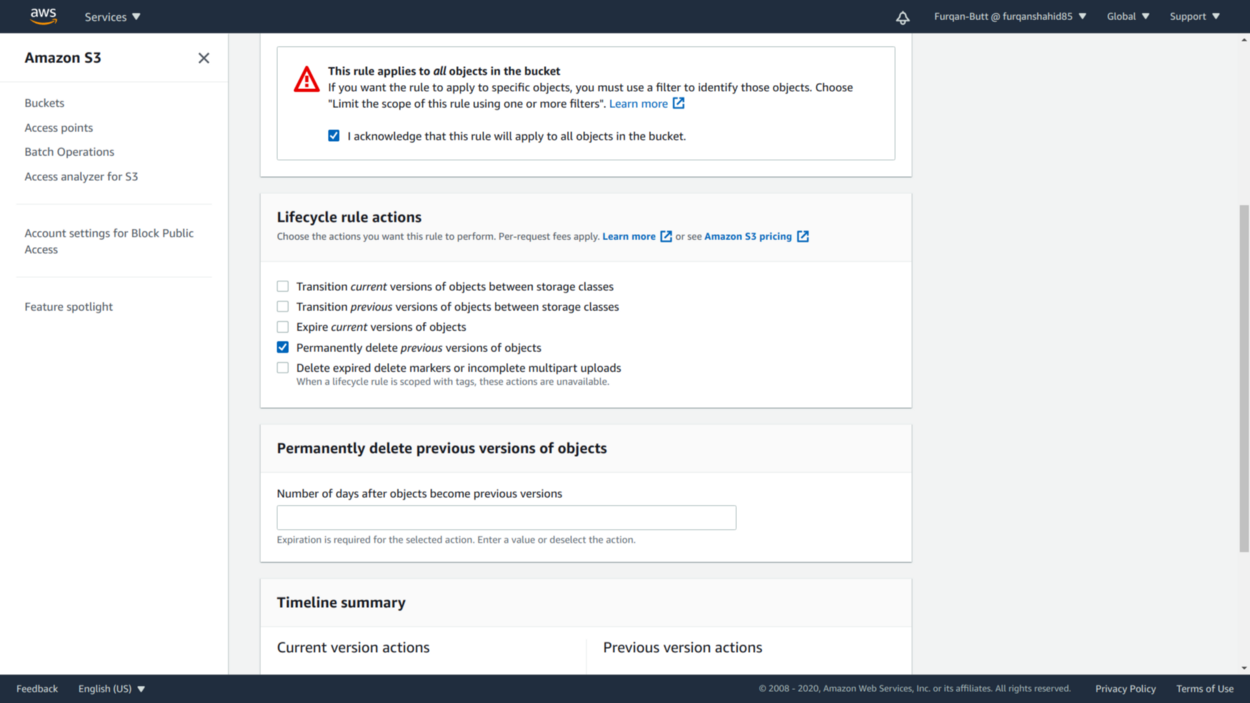
3.a Transitioning current objects versions



3.b Transitioning non-current objects versions

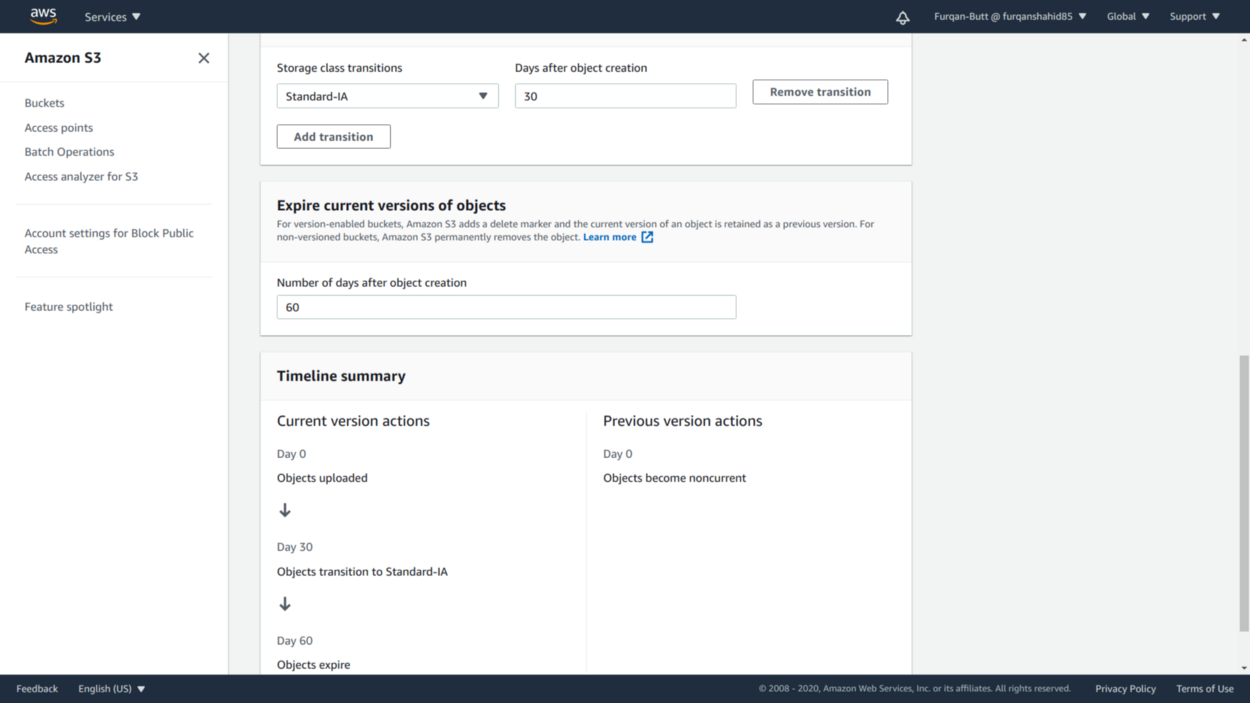


3.c Expiring current versions of objects



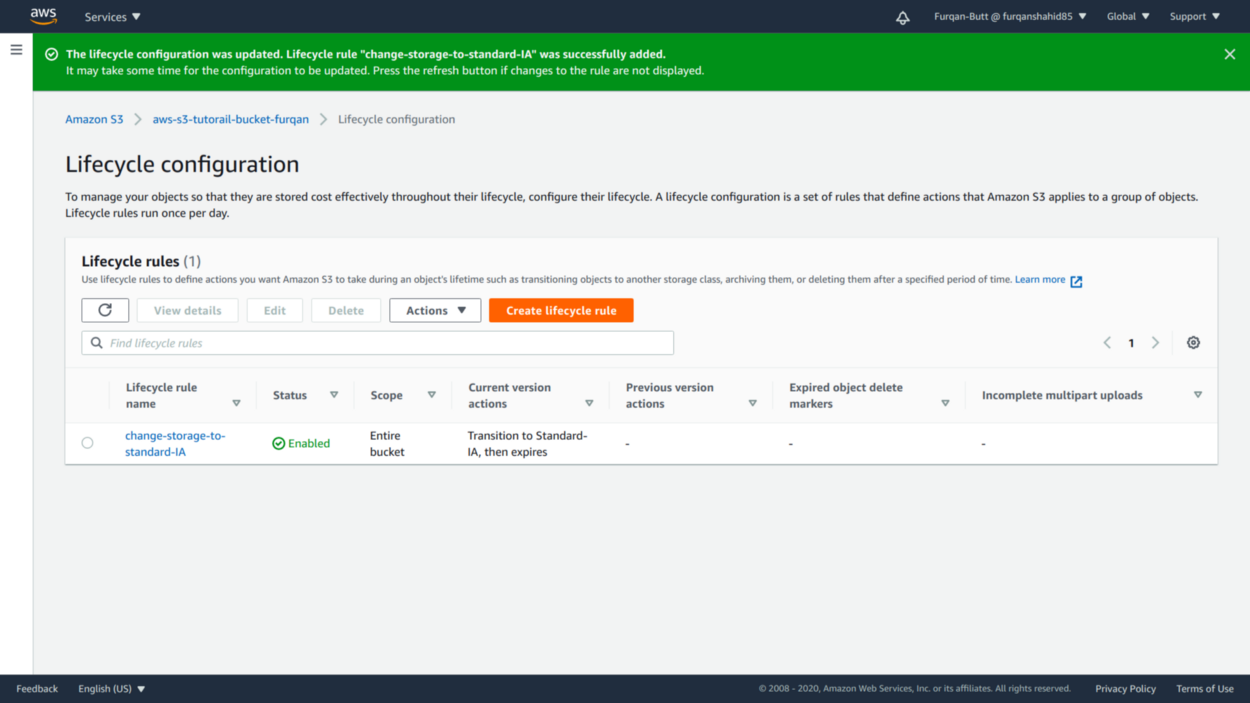
3.d Permanent deletion of objects

* We can see a summary of our life cycle rules. I have set transition current object versions to standard-IA after 30days of upload and expiration after 60 days.



Life cycle rules summary.

* On rule creation, we can see the details of the rules created. We can choose to **edit, delete** or **disable** the rule as well.



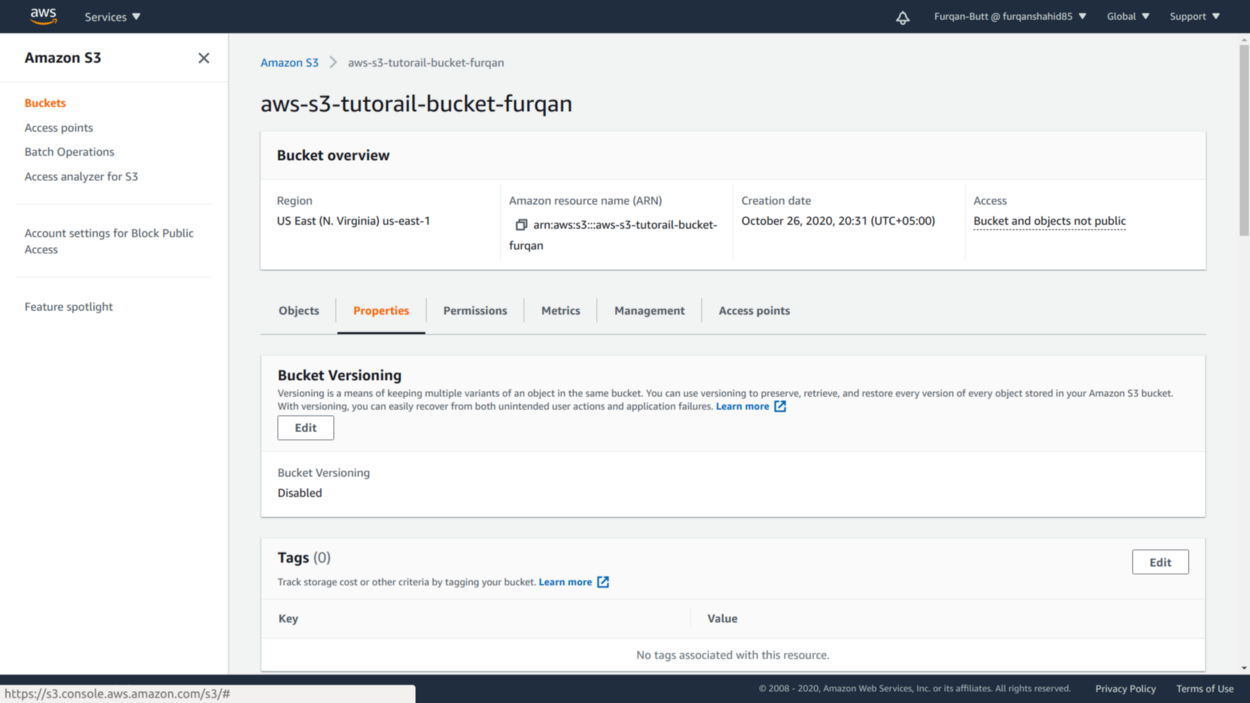
Rules Creation Success.

Exploring Bucket Versioning

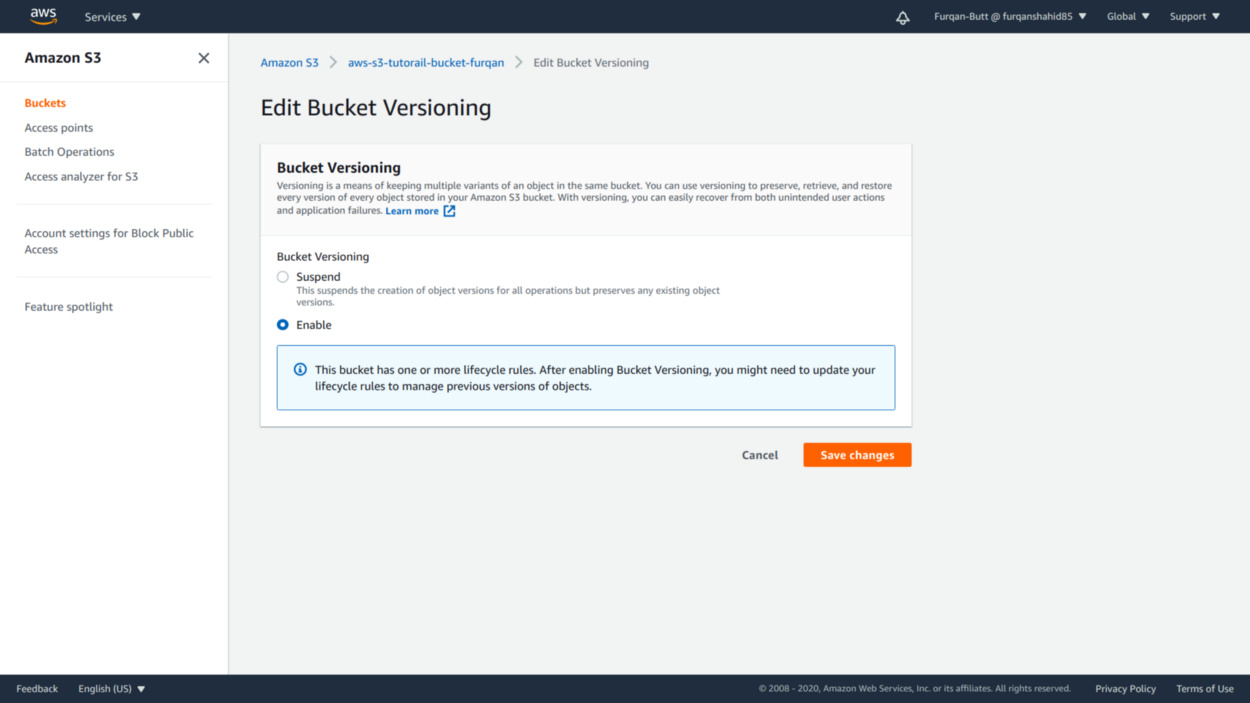
Bucket versioning one of the most prevalent features of AWS S3. Buckets with versioning enabled retain all the versions of an object

Enabling bucket versioning is simple enough on the bucket home, under properties you’ll find the setting for enabling bucket versioning.

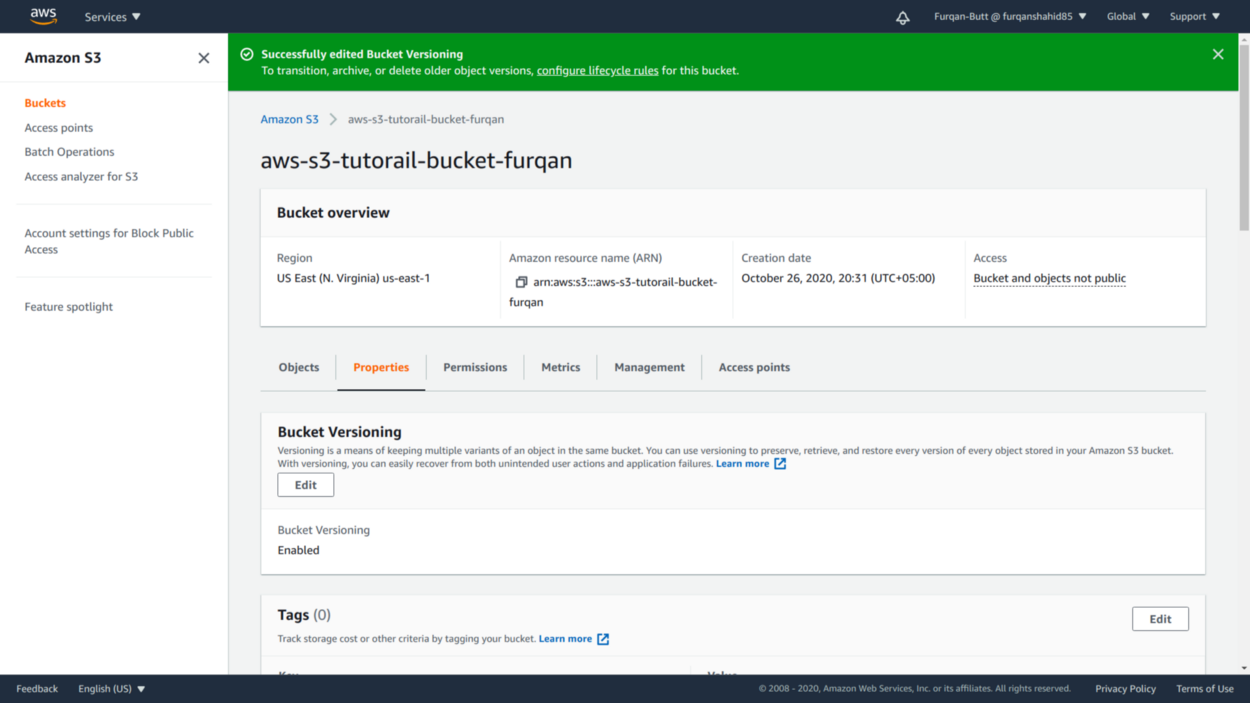
* S3 will maintain all the versions of an object and will display the current version by default.



1.a Enable Bucket versioning

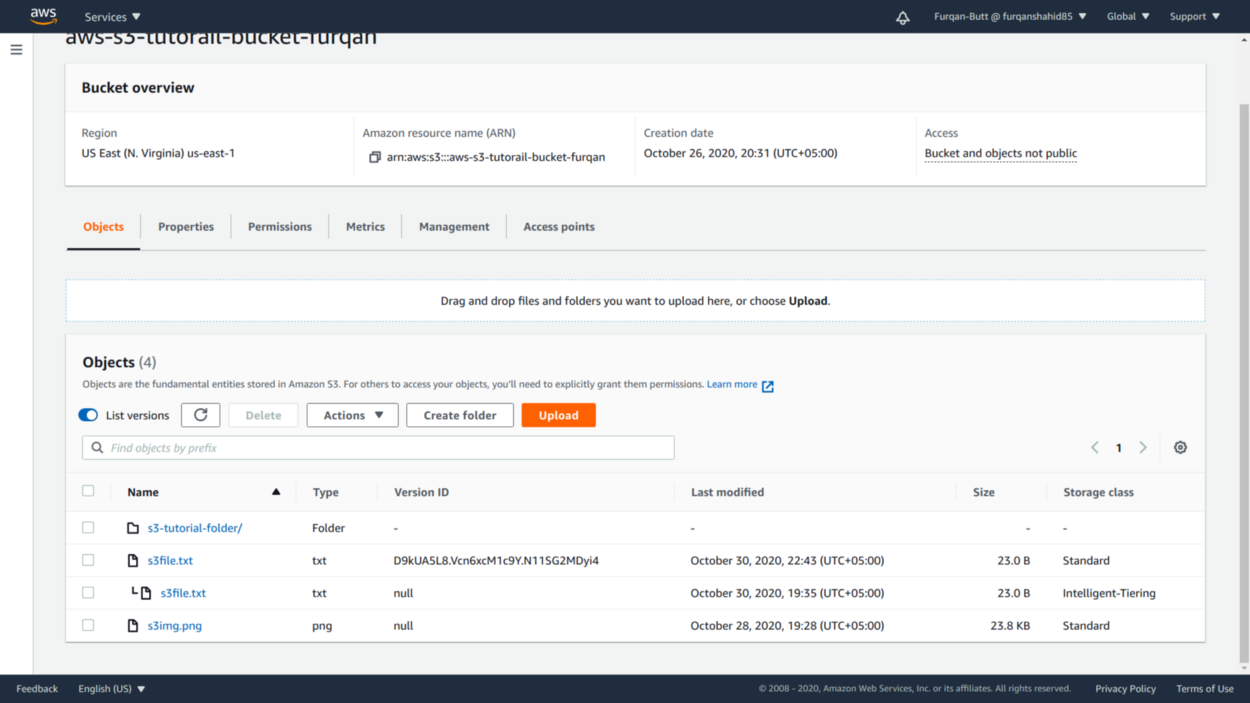


1.b Save Changes.



3 Bucket versioning enabled.

* We can enable bucket versioning under the objects tabs which will list all the versions of the objects.
* Objects uploaded before bucket versioning was enabled are uploaded with null version id.
* New versions are created with standard storage even if the previous versions were under some other storage class.

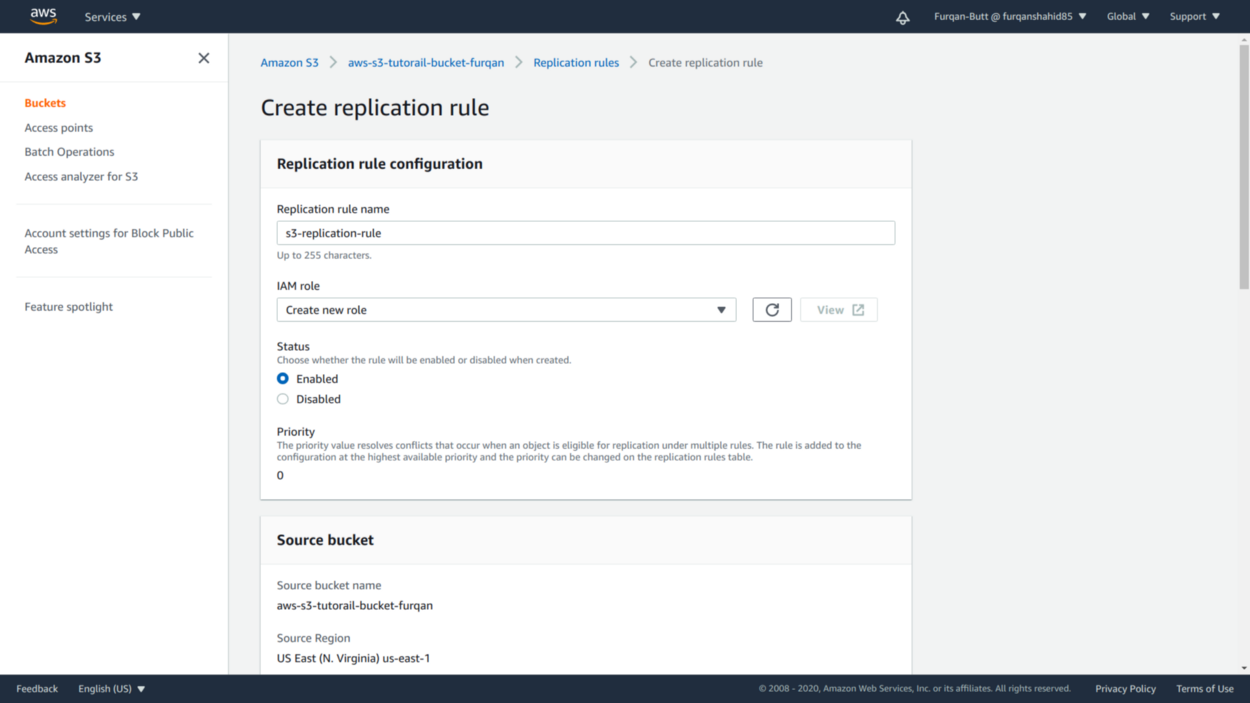


Listing Object Versions.

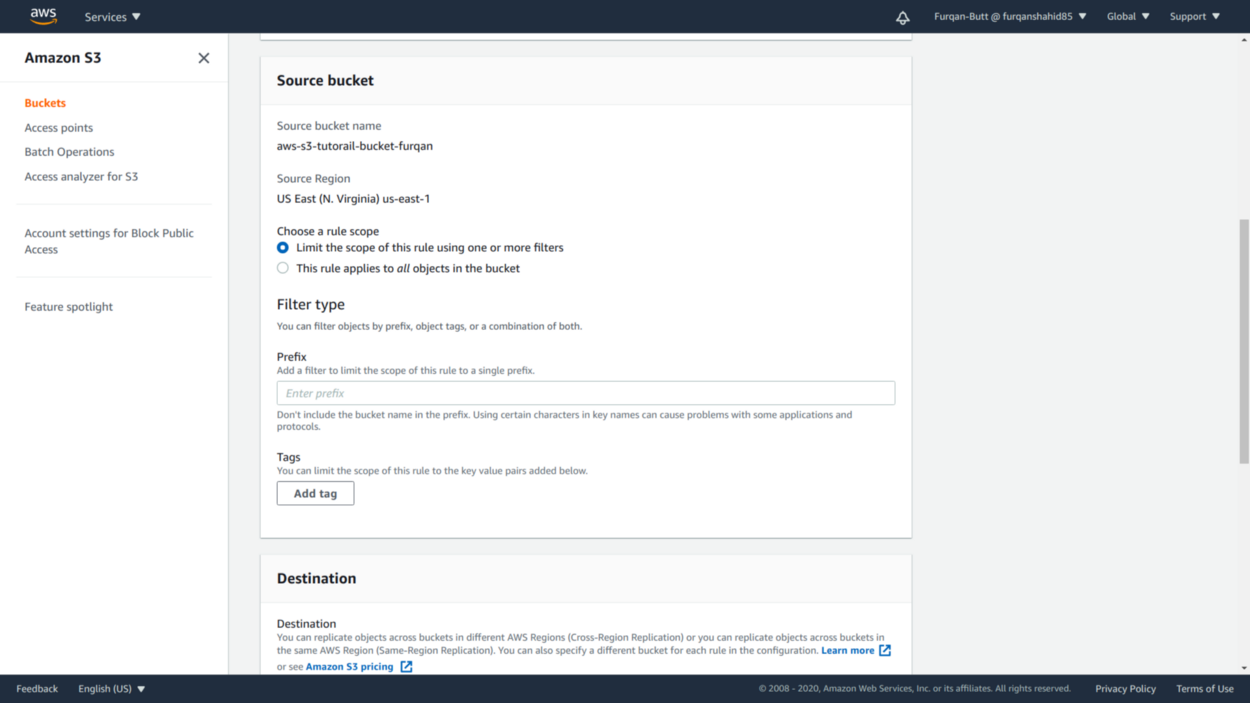
Exploring Object Replication (CRR VS SRR)

S3 also allows replicating objects to a different bucket for backup or some other purposes. Objects can be replicated to bucket with the same region (SRR) or to some other region (CRR).

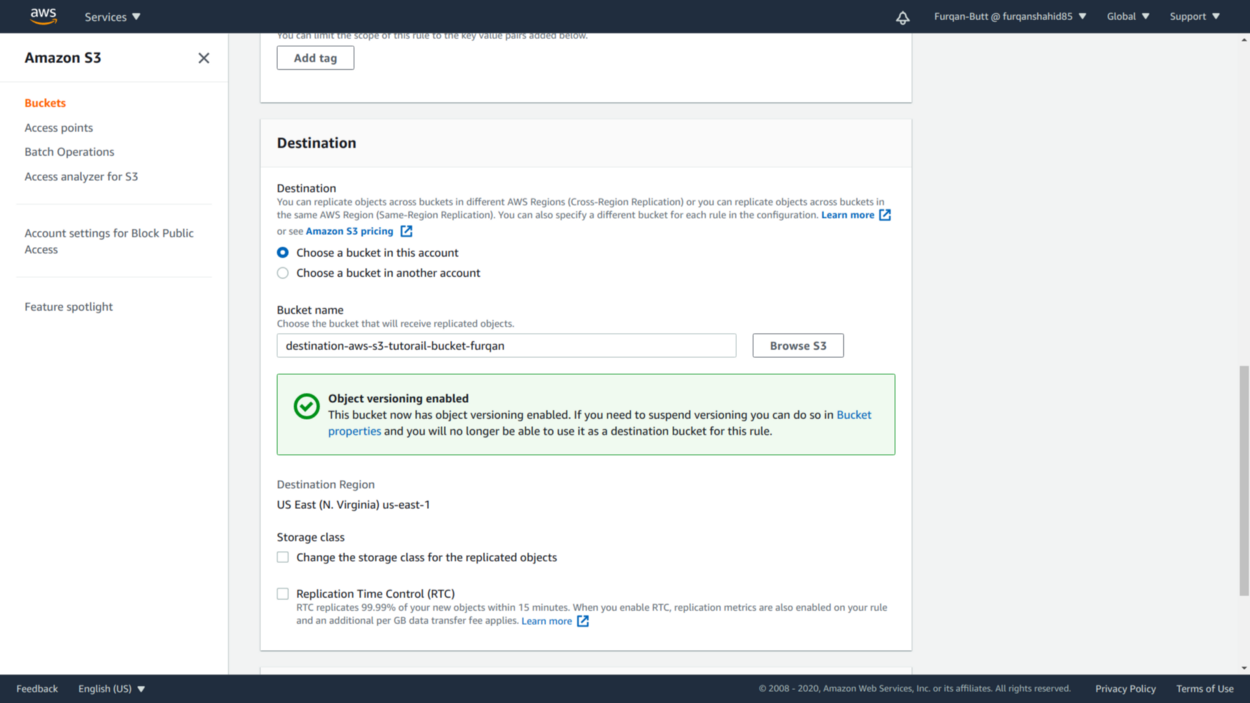
* For replication goto management, under replication rules heading click on **create replication rule**.
* For replication, it is mandatory we **set up an IAM role**to allow S3 to put objects to the destination bucket. Also, bucket versioning for destination bucket must be enabled.



1. Setting up Replication Rule.

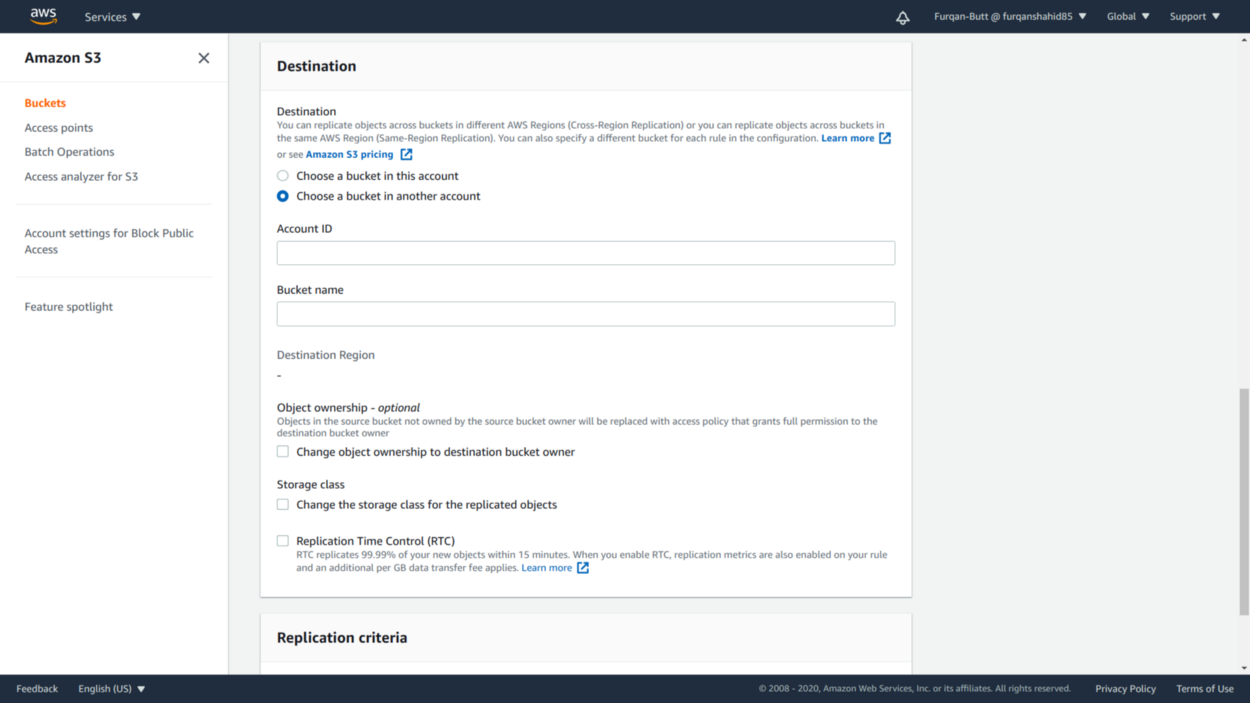


2. Setup optional Filter Conditions.



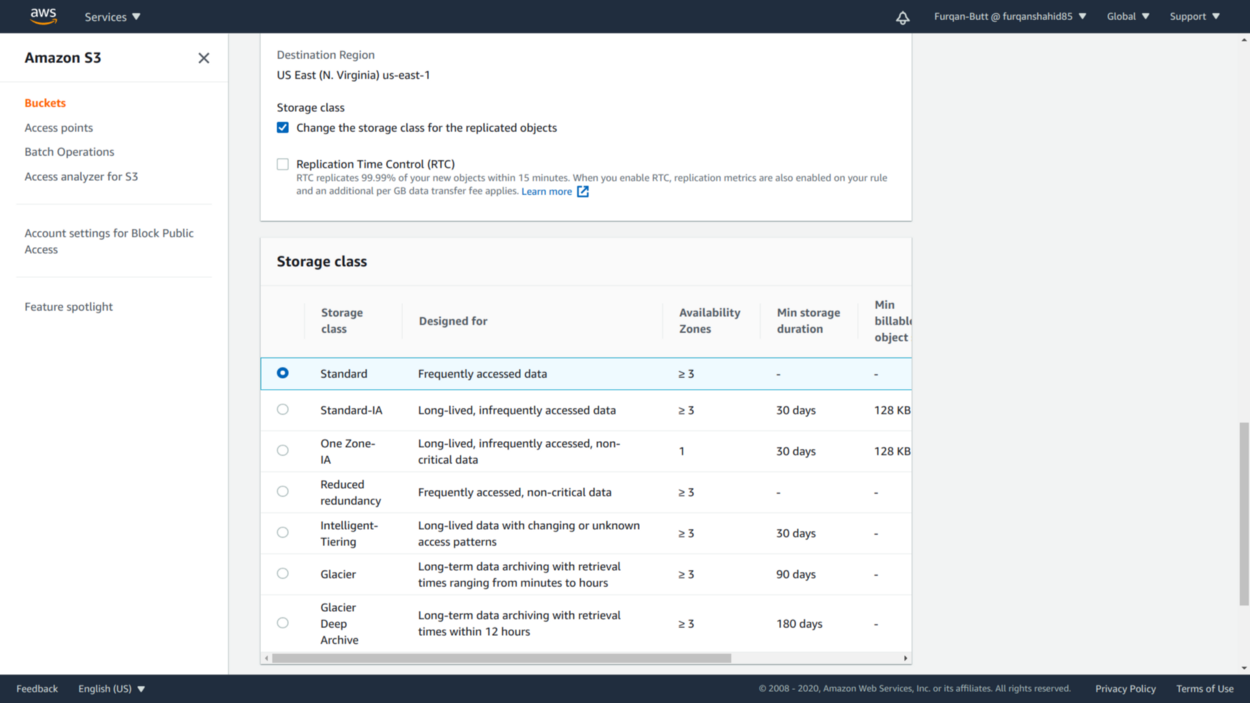
3. a Select Destination Bucket in the same AWS account.

* Account ID of the destination bucket and Bucket name is required for replication to some different AWS account. We can also change bucket ownership via this operation.

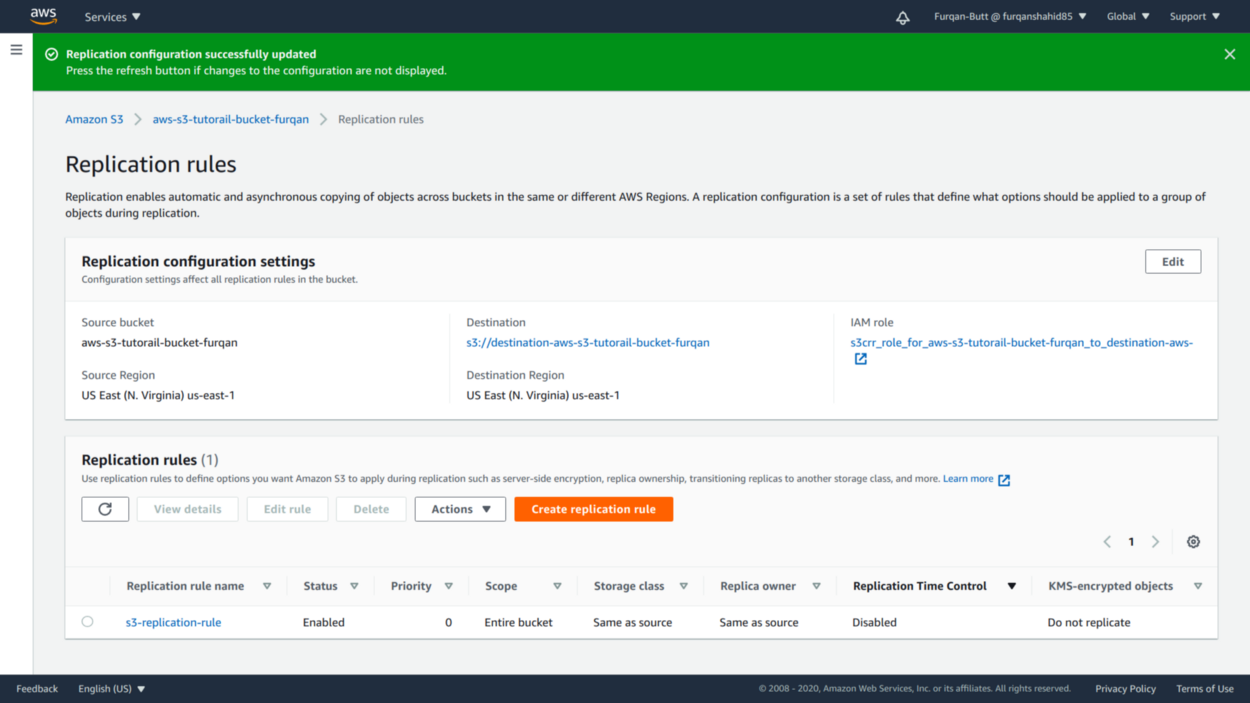


3.b Destination Bucket in some other AWS account.

* We can also change the storage class of the replicated objects in the destination bucket.
* Another option we can enable is replication time control (RTC). Additional charges per GB are charged but 99.99% data is copied within 15 mins.
* We can also set encryption at time of replication.



4 Change objects storage class.



5 Replication Rule creation success.

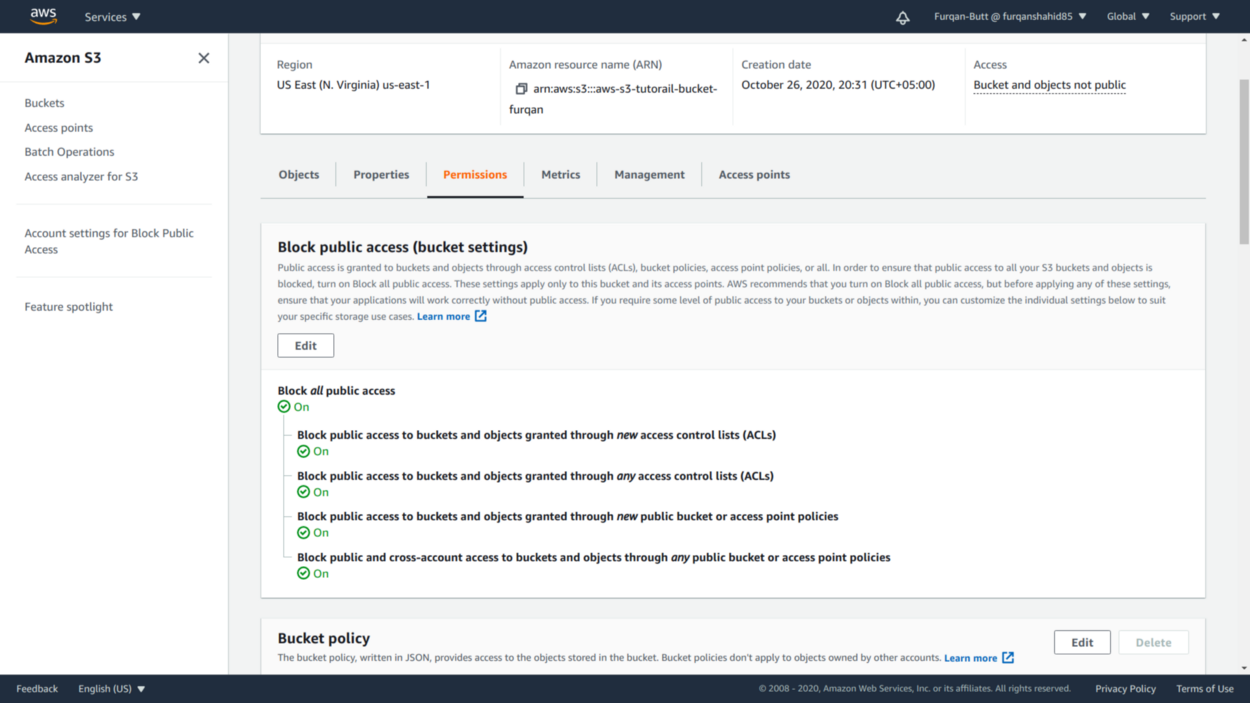
* Lifecycle rules run once a day at midnight (UTC). New lifecycle rules can take up to 48 hours to complete the first run.

Restricting Access to Objects and Buckets

To prevent unintentional overwrites, deletes or unauthorized access S3 offers a number of different approaches to secure objects stored in a bucket. One or more of the following approaches can be used to ensure the security of our objects:

Block Public Access

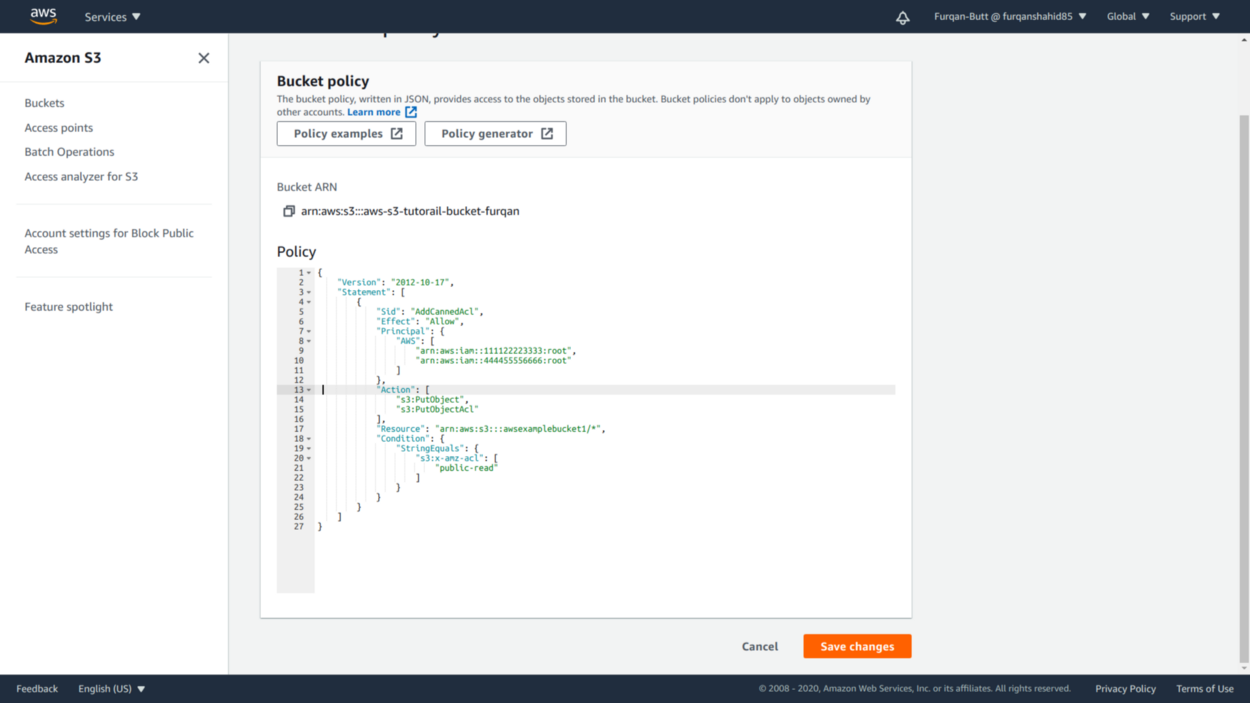
By default buckets are private and can’t be accessed by none other than the owner of the account.



Private bucket.

Bucket Policies

Bucket policies are used to define what kind of access is to be provided to objects with in a bucket. Bucket Policies are JSON based. We can define what kind of actions can be performed and on what objects.



Example Bucket Policy.

IAM roles and policies to Restrict Access

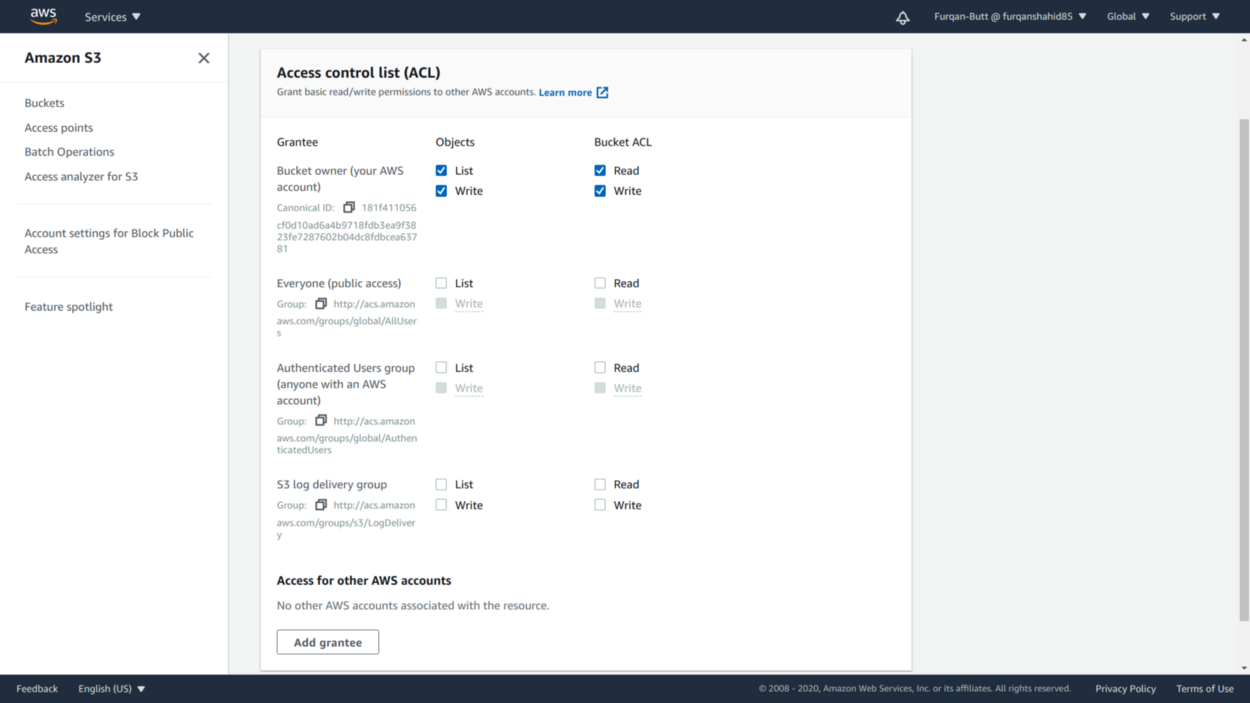
We can restrict which users or specific AWS services to allow or restrict access to objects using the IAM policies and role. These are also JSON based. IAM policies are attached to users, groups or roles.



IAM policy Example

S3 Access control lists (ACLs)

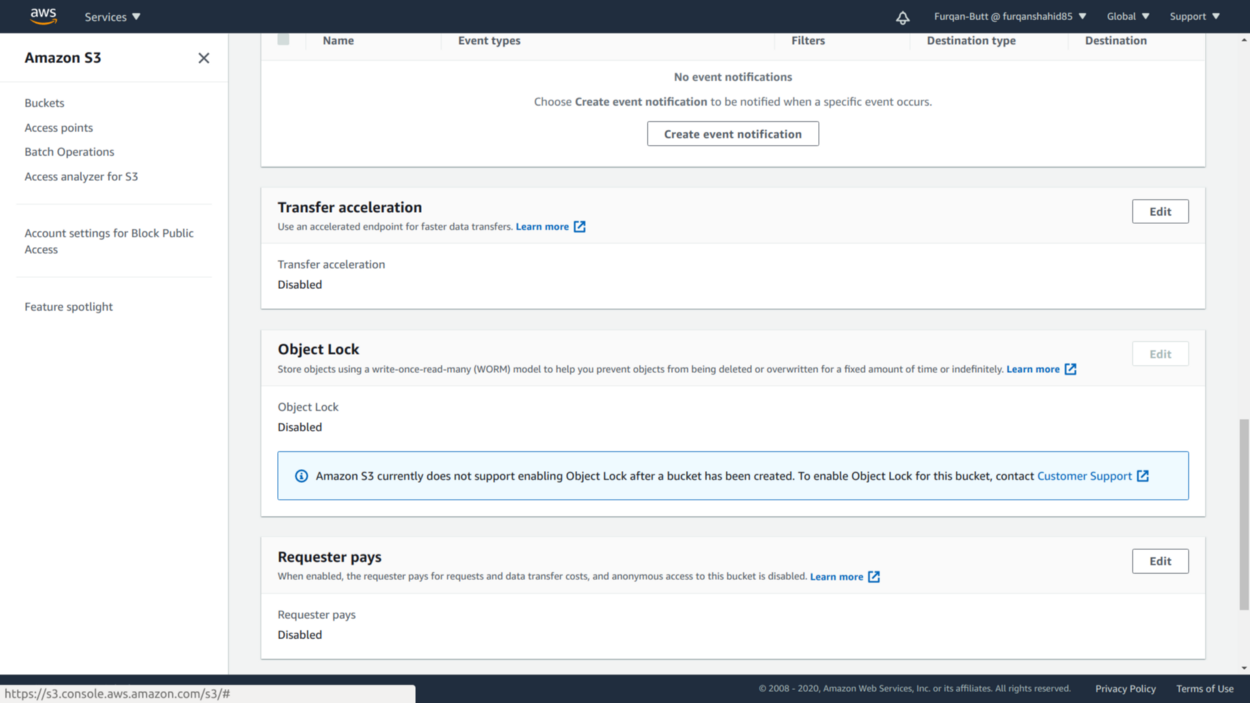
S3 ACLs are another way of restricting access to objects. These are simple read/wrtie access to specific users



S3 ACLs

Object Locks

Object Locks are used to prevent overwrites and deletes once the objects are created. They follow the write-one-read-many (WORM) model. **Objects locks must be set at bucket creation time.**



S3 Object Lock