```
const {
  Aborter,
  BlobURL,
  BlockBlobURL,
  ContainerURL,
  ServiceURL,
  StorageURL,
  SharedKeyCredential,
  uploadStreamToBlockBlob
} = require('@azure/storage-blob');

import stream from 'fs'



export const singleUpload = async (_, args, { pool, request }) => {
  try {

    return args.file.then(async filedata => {

      const { createReadStream, filename, mimetype, encoding } = await filedata;
      const account = 'accName'
      const accountKey = 'acckey'


      // Use SharedKeyCredential with storage account and account key
      const sharedKeyCredential = new SharedKeyCredential(account, accountKey);
      // Use sharedKeyCredential, tokenCredential or anonymousCredential to create a pipeline
      const pipeline = StorageURL.newPipeline(sharedKeyCredential);
      const serviceURL = new ServiceURL(
        // When using AnonymousCredential, following url should include a valid SAS or support public access
        `https://${account}.blob.core.windows.net`,
        pipeline
      );

      const containerName = "mycontainer";
      const containerURL = ContainerURL.fromServiceURL(serviceURL, containerName);
      const blobName = "newblob" + new Date().getTime();
      const blobURL = BlobURL.fromContainerURL(containerURL, blobName);
      const blockBlobURL = BlockBlobURL.fromBlobURL(blobURL);


      const uploadBlobResponse = await uploadStreamToBlockBlob(
        Aborter.timeout(30 * 60 * 60 * 1000),
        createReadStream,
        blockBlobURL,
        20
      );

      console.log(
        `Upload block blob ${blobName} successfully`,
        uploadBlobResponse
      );

      // return { filename, mimetype, encoding }
      // return true
      return "successful upload"
    })
  }
```

```
    catch (error) {
        return error
    }
}
```