

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

[Continue](#)

Node.js



Olamilekan Odukoya [Follow](#)

Oct 30, 2019 · 11 min read



In this relatively short tutorial, I'll be showing you how to upload an image to Google Cloud Storage and also how you can resize the image.

As developers, one of the things we shouldn't do is to always store an Image to our database as this could open up our application to all sort security vulnerabilities, it also eats up unnecessary space in our database since they are stored as a **blob file**, and finally, it could lead to an increase in the cost of backing up data. Solutions to these problems could either be to Store our images in the file system, and create links/pointers to these images in the database but this could be slow when we need to fetch those images. Another way this problem can be solved is to serve our files with a CDN (Content Delivery Network). A Content Delivery network is often used to accelerate the

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

be available to the users until you clear the cache. The best option is to use an online file storage web service on the cloud.

The Cloud is just a remote server sitting somewhere, where you can store your content and access them quickly, easily, securely, and from anywhere in the world. One of the advantages of using the Cloud is because of how elastic, accessible, and secure it can be. A lot of companies offer these cloud-based services including Tech Giants like **Amazon** and **Google**.

In this tutorial, we will be using **Google's cloud-based** service known as **Google Cloud Platform**. Google Cloud platforms offer a range of cloud-based services but the one we are going to be focusing on is going to be *Google Cloud Storage*. Google cloud storage is a service on **GCP** that allows you to store static files like images and files. The advantage of using this is that instead of storing say an image in your database, you instead, upload the image to **GCS** and get a link/url to that file in which you can then store into your database hence, saving disk space in your database and reducing time spent in trying to backup your data.

Getting Started

Before we can start uploading images or any files, we need to first create an account, to do so, copy and paste the URL below into your browser and click **Enter**. This takes you the Google Cloud platform's website.

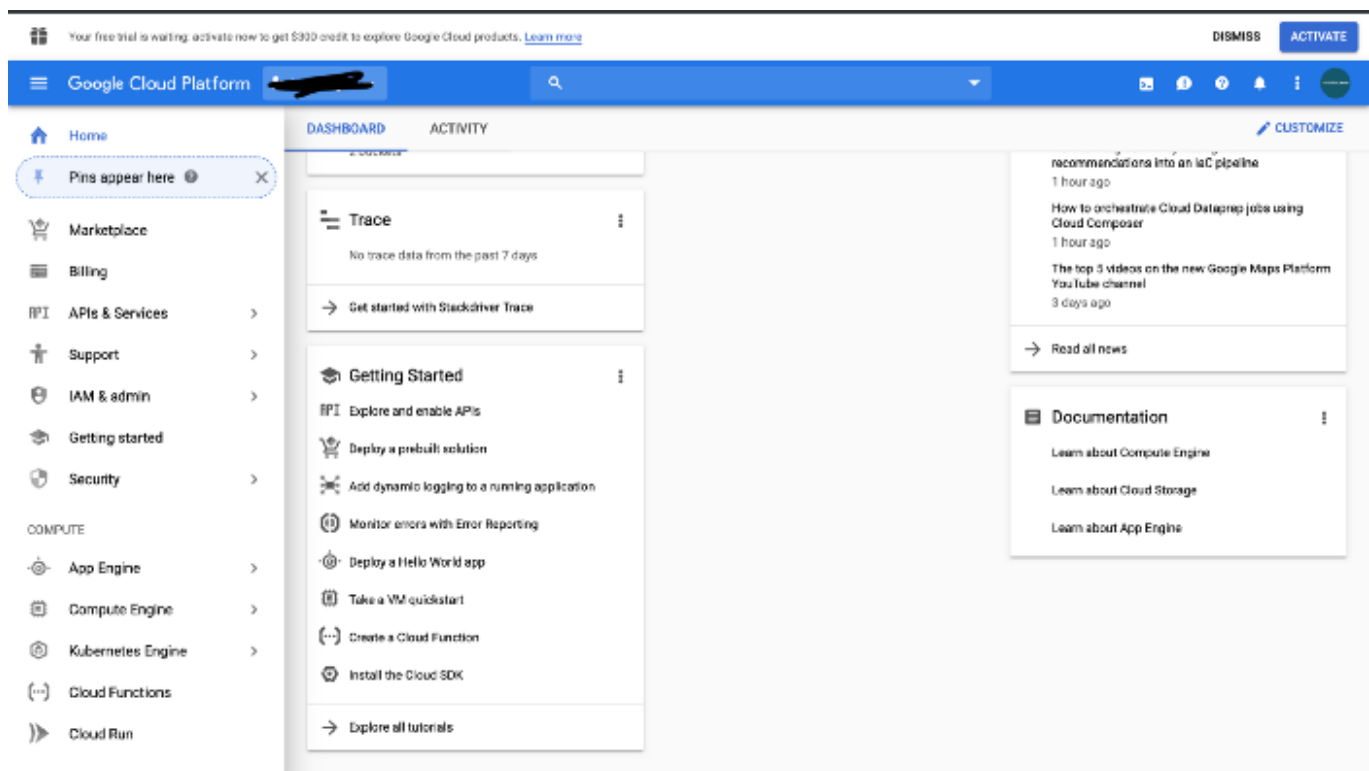
<https://cloud.google.com/storage/>

After the page loads, you should see something similar to what I have below.

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

Click on the **“Go To Console”** Button. This takes you the Cloud Console. Which looks something similar to what we have below.



You'll notice that I painted a part of the picture, click on that part on your screen. This redirects us to a page where we are being asked to create a project. Give it a name and click on the create button. This action creates a new project for us in which we can associate our **GCS** project with.

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue



You have 15 projects remaining in your quota. Request an increase or delete projects. [Learn more](#)

[MANAGE QUOTAS](#)

Project name *

my very own project



Project ID: my-very-own-project-257414. It cannot be changed later. [EDIT](#)

Location *

No organization

[BROWSE](#)

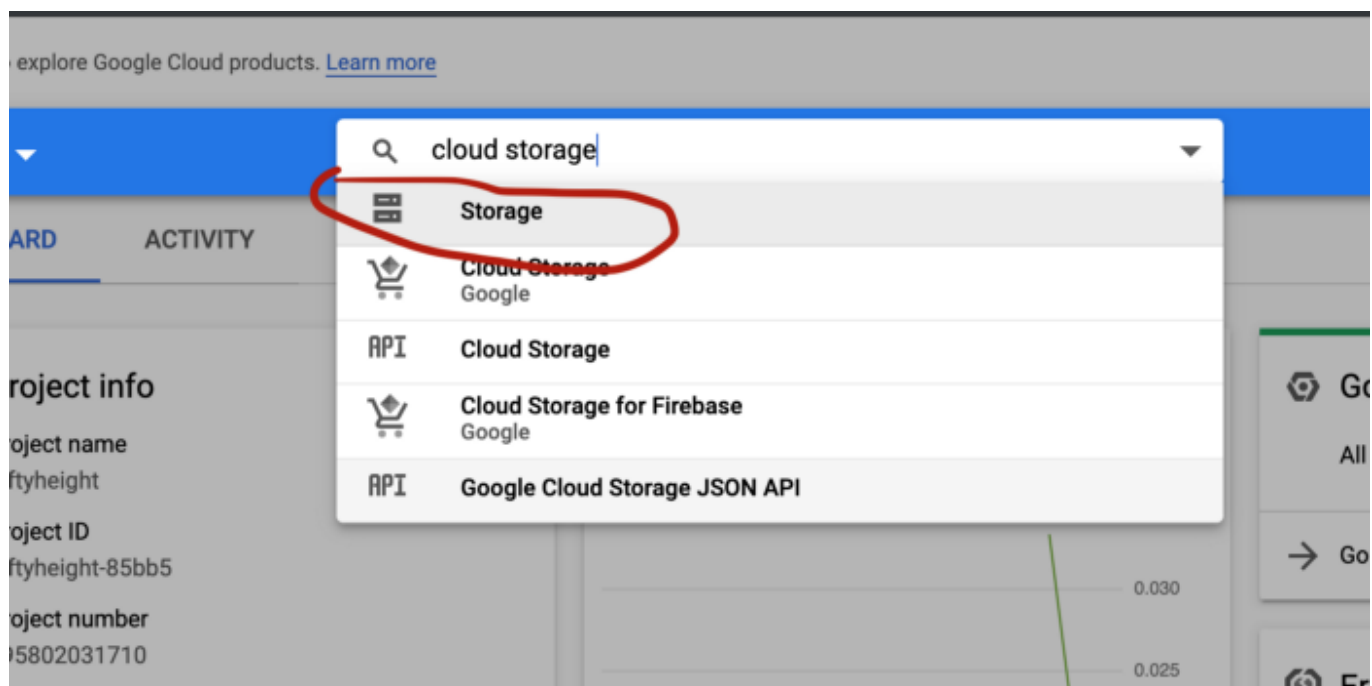
Parent organization or folder

CREATE

CANCEL

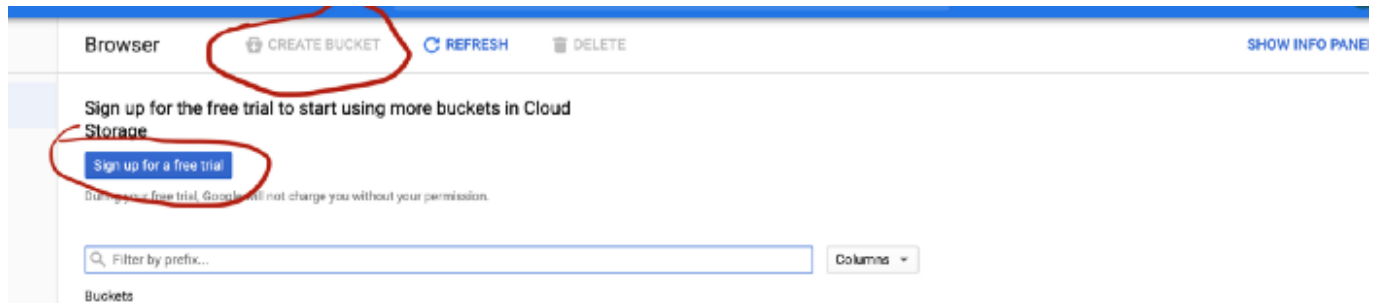
Yours might be way different from mine if it's your first using **GCP**.

In the search bar, type **Cloud Storage**.



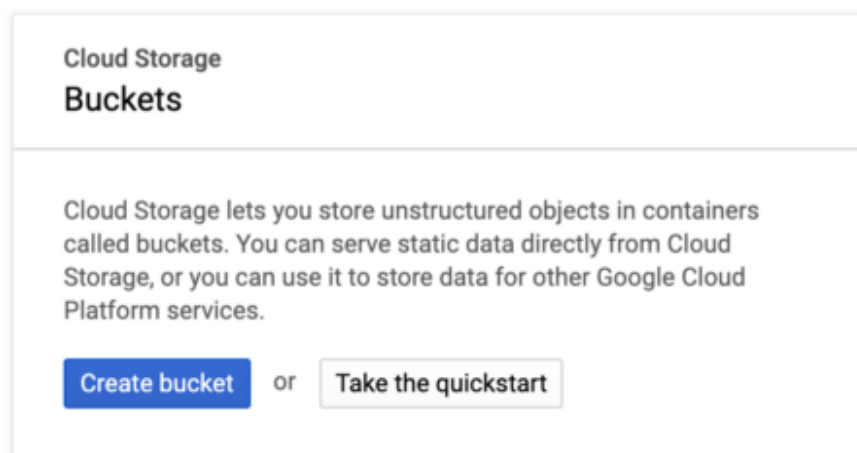
We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue



You'll notice that the "Create Bucket" button isn't active, and also there is a button that is asking us to "Signup for free". Click on the "Sign Up Button" and fill all the necessary information. You might also be asked to enter your Credit/Debit Card Details before you can be able to complete the "Sign Up exercise".

After you are done, the "Create Bucket" button should be active, click on it, you should see a modal similar to what we have below.



We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

to the last select box. Finally, click on the *create* button.

- ✓ Choose where to store your data
- ✓ Choose a default storage class for your data
- ✓ Choose how to control access to objects
- Advanced settings (optional)

Encryption

- ☒ Google-managed key
No configuration required
- ☐ Customer-managed key
Manage via Google Cloud Key Management Service

Retention policy

Set a retention policy to specify the minimum duration that this bucket's objects must be protected from deletion or modification after they're uploaded. You might set a policy to address industry-specific retention challenges. [Learn more](#)

☐ Set a retention policy

Labels

Labels are key:value pairs that allow you to group related buckets together or with other Cloud Platform resources. [Learn more](#)

+ ADD LABEL

CREATE

CANCEL

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

[Continue](#)

New Project



You have 15 projects remaining in your quota. Request an increase or delete projects. [Learn more](#)

[MANAGE QUOTAS](#)

Project name *

my very own project



Project ID: my-very-own-project-257414. It cannot be changed later. [EDIT](#)

Location *



No organization

[BROWSE](#)

Parent organization or folder

[CREATE](#)

[CANCEL](#)

Click on the **create** button to finally create a **Bucket**. A **Bucket** is basically a place on the cloud where all of your data are being stored. According to the Google Cloud documentation, a **Bucket** is said to be a “Container that holds your data.” To read more about **Buckets** visit the link below.

<https://cloud.google.com/storage/docs/key-terms>

That being done, we can now move to our development environment and start writing some code!

Setting Up The Development Environment

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

if you are using **Visual Studio Code** as your Code Editor you can type the command below after the initial ones to open up the Editor.

```
code .
```

Once you are in your directory and have confirmed, type in the command below to initialize your project.

```
npm init --y
```

The command below is often used whenever a new project is to be started in **Node.js**. This allows us to keep track of all install dependencies when have used in developing the project. The result of running the command is shown below. A **json file** with some **meta-data** about our project.

```
{
  "name": "uploadit",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```


We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

```
npm i express body-parser multer
```

We need **express** for creating a server that is going to power our application, **body-parser** for parsing requests whenever it hits our server, and **multer** for parsing files which must have been part of the request object.

After successfully completing the action, a **node_modules** directory is being included in your project directory. This folder contains all the dependencies our application needs for it to be able to run.

Creating A Server

After you are done, in your root directory create a file and name it **index.js**. This is where we are going to put the code which is going to create a server for our application. At this rate, your application's project structure should look similar to the picture below.

In your **index.js** file, paste the code below into it.

```
const express = require('express')
const bodyParser = require('body-parser')
const multer = require('multer')

const app = express()

app.listen(9001, () => {
  console.log('app now listening for requests!!!')
})
```

After you're done with that, run the command below to start your server.

```
node index
```

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

[Continue](#)

the app when we have to manually kill the server, and then restart before it can be able to notice any changes in our application. A solution to this problem is to install **nodemon** which we can use to automatically start our project anytime we make a change to any file in our project. To install it run the command below in your terminal.

```
npm i -D nodemon
```

This installs it as a **dev-dependency** in the project. Now, go to your **package.json** file and change it's contents to what is in the snippet below.

```
1
2  {
3    "name": "uploadit",
4    "version": "1.0.0",
5    "description": "",
6    "main": "index.js",
7    "scripts": {
8      "test": "echo \"Error: no test specified\" && exit 1",
9      "start": "nodemon index.js"
10   },
11   "keywords": [],
12   "author": "",
13   "license": "ISC",
14   "dependencies": {
15     "body-parser": "^1.19.0",
16     "express": "^4.17.1",
17     "multer": "^1.4.2"
18   }
19 }
```

package.json hosted with ♥ by [GitHub](#)

[view raw](#)

You'll notice that in the **script** key, a new set of **key-value** pair has been added to it. Now, go to your terminal and run the command below to start your application server with

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

You should see something similar to what we have below.

```
> uploadit@1.0.0 start /Users/new/Desktop/Play-ground
> nodemon index.js

[nodemon] 1.19.4
[nodemon] to restart at any time, enter `rs`
[nodemon] watching dir(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
app now listening for requests!!!
```

Setting Up A Service Account Key

Now that we have a server running, we can now start making requests in order to push an image or file into our **bucket**. But to do so, we need a dependency and also some certain credentials from our **bucket** to do so.

In your **terminal**, type in the command to install the **google cloud storage** dependency which is going to help us with the image upload.

```
npm i @google-cloud/storage
```

Create a folder in the root of your project and name it **assets**. This is where we are going to put the files and images we want to push into our bucket. Once you are done with that, put a random picture in it and go back to the **GCP** console. We need to create a **service account key** which **gcs** is going to use to authorize us to **put** a file in our **bucket**.

On the **GCP** console, search for **API**, click on the **api & services** option. You should be redirected to a new page. Scroll down and click on the **storage link**

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

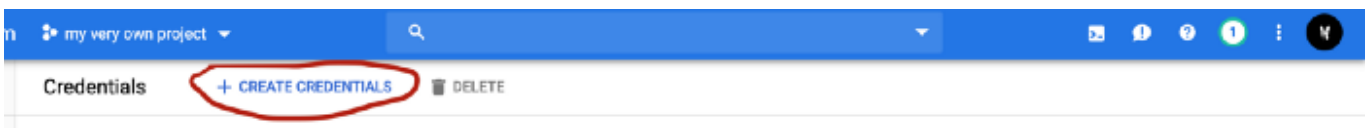
☐ Hide unused APIs ?

Filter

Name	↓ Requests	Errors (%)	Latency, median (ms)	Latency, 95% (ms)
BigQuery API				
BigQuery Storage API				
Cloud Datastore API				
Cloud SQL				
Cloud Storage				
Google Cloud APIs				
Google Cloud Storage JSON API				
Service Management API				
Service Usage API				
Stackdriver Debugger API				
Stackdriver Logging API				
Stackdriver Monitoring API				
Stackdriver Trace API				

Rows per page: 50 1 - 13 of 13 < >

After clicking on the link, you will be redirected to a new page. At the left section of the page, click on the **credentials** button, afterward, click on the **add credentials** button as shown below.



Then, click on the **service account option**. Fill in the forms and then click on the **create button** as shown below.

Create service account

- 1 Service account details
- 2 Grant this service account access to project (optional)

Service account details

Service account name

randomname

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

Service account description

a little tut on image upload

Describe what this service account will do

CREATE

CANCEL

You should be redirected to the *create role* section. Scroll down or search for *storage*. Select it and also select *API KEYS ADMIN*.

Service account permissions (optional)

Grant this service account access to my very own project so that it has permission to complete specific actions on the resources in your project. [Learn more](#)

Select a role

Type to filter

Service Management

Service Networking

Service Usage

Source

Stackdriver

Stackdriver Debugger

Stackdriver Profiler

Storage

API Keys Admin

API Keys Viewer

Service Usage Admin

Service Usage Consumer

Service Usage Viewer

API Keys Admin

Ability to create, delete, update, get and list API keys for a project.

MANAGE ROLES

Once that's done, you are once again being redirected to another page. Just click on the *create key* button as shown below.

Grant users access to this service account (optional)

Filter tree

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

Grant users the permission to administer this service account

Create key (optional)

Download a file that contains the private key. Store the file securely because this key can't be recovered if lost. However, if you are unsure why you need a key, skip this step for now.

+ CREATE KEY

Click on create

my very own project

Create service account

Service account details — Grant this service account access to project (optional)

3 Grant users access to this service account (optional)

Grant users access to this service account (optional)

Grant access to users or groups that need to perform actions as this service account. [Learn more](#)

Service account users role ?
Grant users the permissions to deploy jobs and VMs with this service account

Service account admins role ?
Grant users the permission to administer this service account

Create key (optional)

Download a file that contains the private key. Store the file securely because this key can't be recovered if lost. However, if you are unsure why you need a key, skip this step for now.

+ CREATE KEY

Create key (optional)

Download a file that contains the private key. Store the file securely because this key can't be recovered if lost. However, if you are unsure why you need a key, skip this step for now.

Key type

☒ JSON
Recommended

☐ P12
For backward compatibility with code using the P12 format

CREATE CANCEL

After it is being created, an automatic download of the account key is being automatically downloaded. *You need to keep this file very safe.*

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

[Continue](#)

CLOSE

Create key (optional)

Going Deeper!

In the previous section, we created a *service account key* and also downloaded it on our machine. So, we need to make use of this file before we can be able to do anything. In your root directory create a folder and name it *config*. Copy and then paste the *JSON* file we downloaded into it. Also, in the *config* directory, create an *index.js* file and paste the code below into it.

```
1
2  const Cloud = require('@google-cloud/storage')
3  const path = require('path')
4  const serviceKey = path.join(__dirname, './keys.json')
5
6  const { Storage } = Cloud
7  const storage = new Storage({
8    keyFilename: serviceKey,
9    projectId: 'your project id',
10  })
11
12  module.exports = storage
```

helpers.js hosted with ❤ by GitHub

[view raw](#)

What we have done here is basically a configuration of the *@google-cloud/storage* dependency we installed earlier on in the project.

After doing this we can now start with the core functionality of our application, which is *image/file* upload.

In the root directory create a folder and name it *helpers*. Then, create a *helpers.js* file in it. We are going to use this file to create some helper functions that would help us with

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

[Continue](#)

```
2  const express = require('express')
3  const bodyParser = require('body-parser')
4  const multer = require('multer')
5
6  const app = express()
7
8  const multerMid = multer({
9    storage: multer.memoryStorage(),
10   limits: {
11     fileSize: 5 * 1024 * 1024,
12   },
13 })
14
15 app.disable('x-powered-by')
16 app.use(multerMid.single('file'))
17 app.use(bodyParser.json())
18 app.use(bodyParser.urlencoded({extended: false}))
19
20 app.post('/uploads', (req, res, next) => {
21 })
22
23 app.listen(9001, () => {
24   console.log('app now listening for requests!!!')
25 })
26
27
```

app.js hosted with ❤ by GitHub

[view raw](#)

You'll notice that our app has changed. We actually haven't done anything much, all we have done is just to use the other dependencies as a middleware in our application.

Proceed to the *helpers.js* file and paste the code snippet into it.

```
1
2  const util = require('util')
3  const gc = require('./config/')
4  const bucket = gc.bucket('all-mighti') // should be your bucket name
```


We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

[Continue](#)

```
11  * - It accepts an object as an argument with the
12  *   "originalname" and "buffer" as keys
13  */
14
15  export const uploadImage = (file) => new Promise((resolve, reject) => {
16    const { originalname, buffer } = file
17
18    const blob = bucket.file(originalname.replace(/ /g, "_"))
19    const blobStream = blob.createWriteStream({
20      resumable: false
21    })
22    blobStream.on('finish', () => {
23      const publicUrl = format(
24        `https://storage.googleapis.com/${bucket.name}/${blob.name}`
25      )
26      resolve(publicUrl)
27    })
28    .on('error', () => {
29      reject(`Unable to upload image, something went wrong`)
30    })
31    .end(buffer)
32  })
```

helpers.js hosted with ❤ by GitHub

[view raw](#)

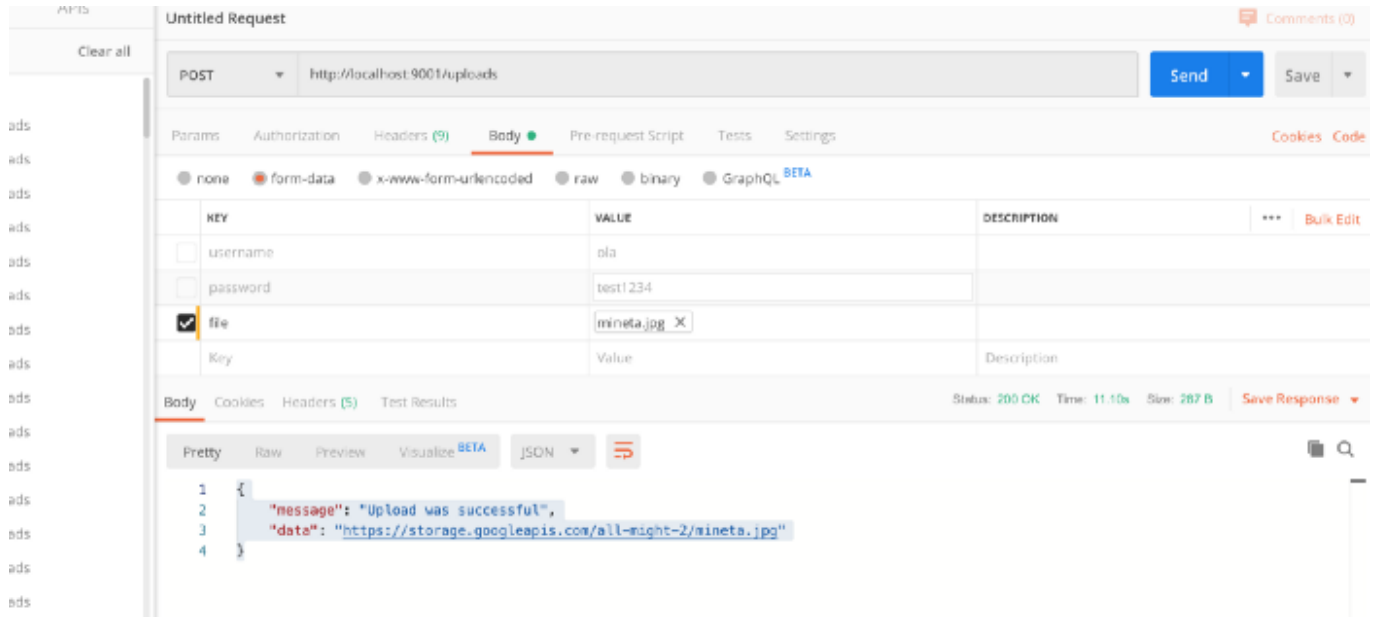
What we have done here is to first import the `@google-cloud/storage` we initially configured and then we proceeded in linking it to our bucket. Afterward, we created a function that returns a *Promise*. The function expects an object which contains the *blob* object. We then *extracted* the file name and the file (*buffer*) from the object. We clean it up, then, create a *read stream*. Once it is done writing the stream, it returns a URL to the image.

Once that is done, go back to your *index.js* file and replace the existing code with the code snippet below.

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

`http://localhost:9001/uploads`



if that's successful, you should see a link to the image.

However, at this rate, the image would not be publicly available. You need to go back to the *gcs* console and *edit permission* of the file.

Go to *permissions* => *Add Members* => *storage object viewer*.

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

Continue

Once that is complete, the file can now be publicly available.

Conclusion

In this relatively short tutorial (*lol*) we can see that **gcs** have made file upload easier. With few lines of code, you already have your file uploaded. It also has a very secure system for authorized users who can have access to the files in your bucket.

The application's source code can be found...

<https://github.com/olamilekan000/uploadit>

Also, if you'd like to ask me further questions, feel free to hit me up on Twitter.

https://twitter.com/Principino__

To learn more about **gcs**, check out the docs here

<https://googleapis.dev/nodejs/storage/latest/index.html>

We've made changes to our [Terms of Service](#) and [Privacy Policy](#). They take effect on September 1, 2020, and we encourage you to review them. By continuing to use our services, you agree to the new Terms of Service and acknowledge the Privacy Policy applies to you.

[Continue](#)

Arigato!

[JavaScript](#)[Cloud Computing](#)[Google Cloud Platform](#)[Nodejs](#)[Firebase](#)[About](#) [Help](#) [Legal](#)

Get the Medium app

