

precase__week4

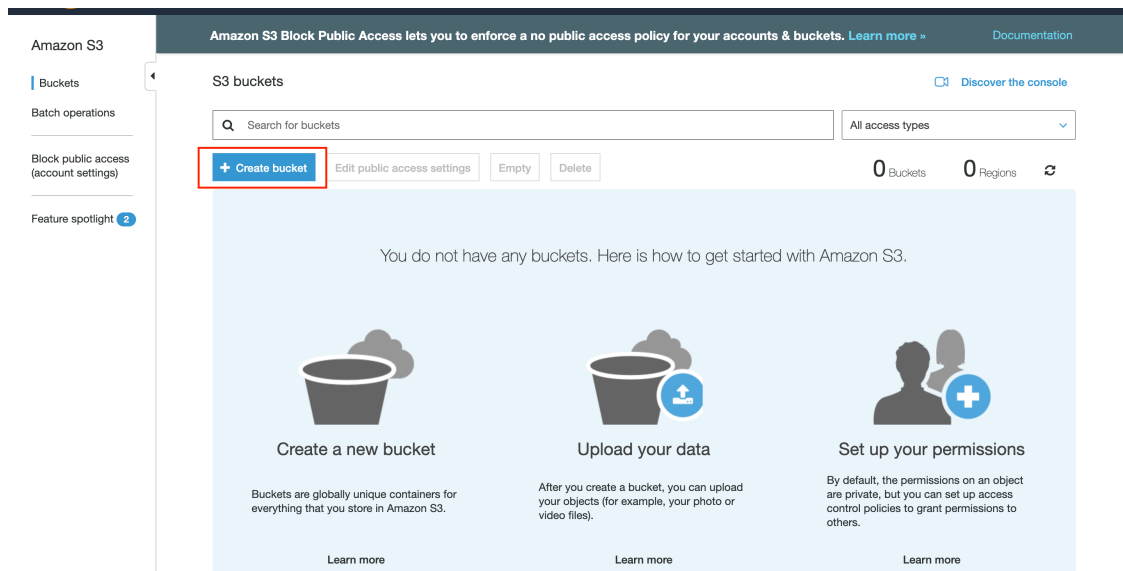
May 13, 2020

0.1 Uploading Files to AWS S3 and setting up API Keys

In this section, our focus is to upload the data available in the zip to AWS S3, and then generate the API Keys (Access Key ID and Secret Access Key) to be used in the case 5.1

0.2 Create S3 Bucket

Login to your AWS portal and then to S3. In order to upload the file, you will need to create a bucket to hold the data files. In your S3 portal, click on the **Create Bucket** button.



Once the Create bucket form opens up, Provide a name for your bucket and click on the create button.

Create bucket

1 Name and region 2 Configure options 3 Set permissions 4 Review

Name and region

Bucket name ⓘ
case-5-1

Region
US West (Oregon) ▾

Copy settings from an existing bucket
You have no buckets0 Buckets ▾

Create Cancel Next

Once the bucket is created, click on the bucket name to go to the bucket detail page. Once in the detail page, click on **Upload** button present in the page.

Amazon S3 > case-5-1

Overview Properties Permissions Management

Upload + Create folder Download Actions ▾

US West (Oregon) ↻

This bucket is empty. Upload new objects to get started.

Upload an object

Buckets are globally unique containers for everything that you store in Amazon S3.

[Learn more](#)

Set object properties

After you create a bucket, you can upload your objects (for example, your photo or video files).

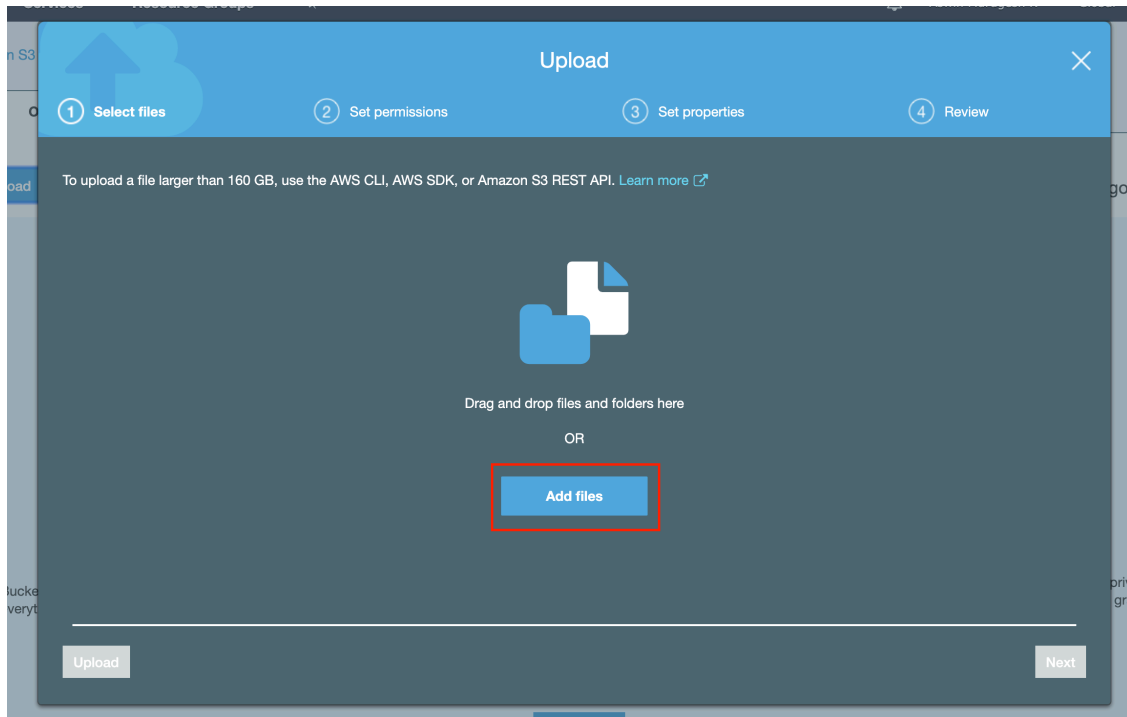
[Learn more](#)

Set object permissions

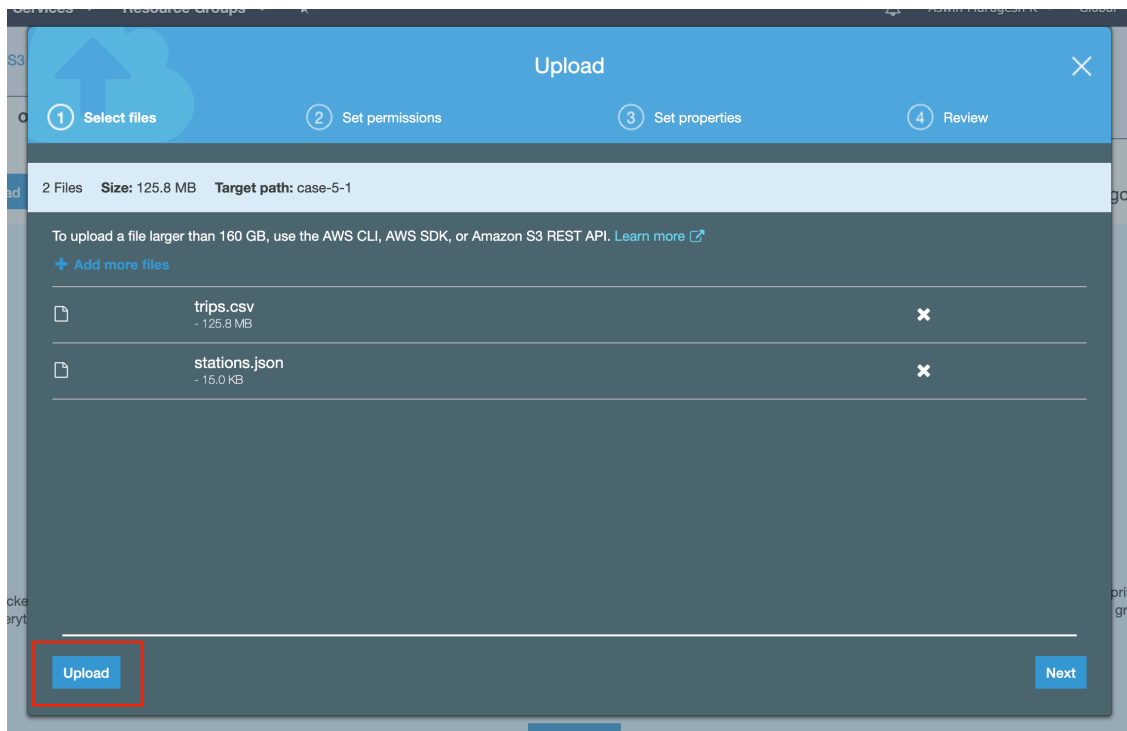
By default, the permissions on an object are private, but you can set up access control policies to grant permissions to others.

[Learn more](#)

When the add form shows up, click on the **Add Files** button



When the dialog box opens up, select the `trips.csv` and `stations.json` file provided to you. You will then see the below page

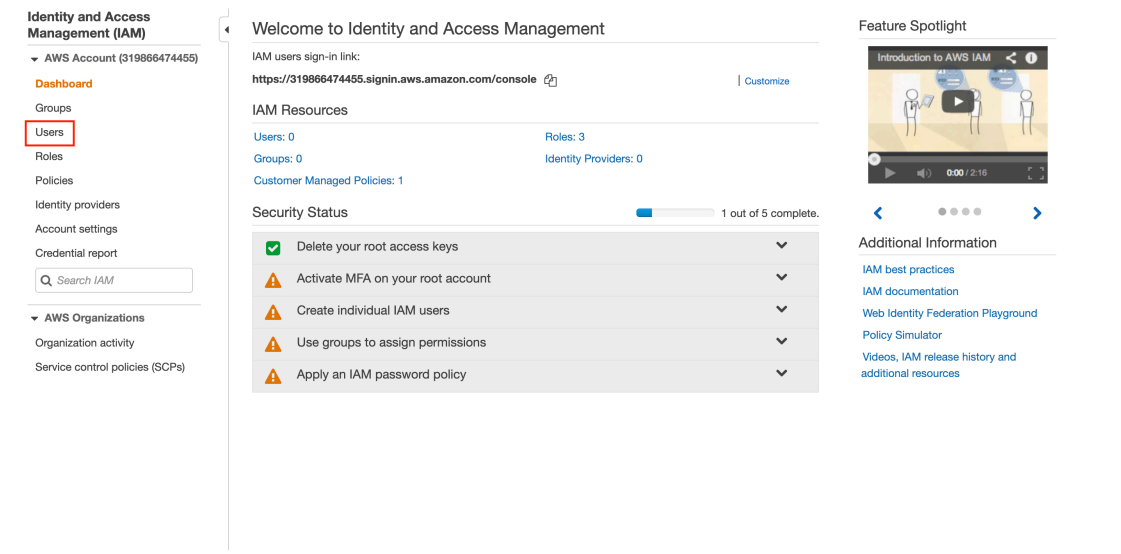


Once you click on the **Upload** button, you can see the **Operation in Progress** button at the bottom of the screen. The data will be uploaded and in a while you will see the two files in the list

of files in the bucket, as below:

0.3 Setup Security Credentials in IAM

Once the files are uploaded, we will need a way to access the uploaded file programmatically. In order to access the files via any programmatic manner (e.g. Python program), you will need to create Security credentials. For that, you will need to go to the IAM service of AWS. Go to **Services** -> **IAM** to see the below page



Click on the Users section to see the list of users in your account. Click on **Add Users** button at the top of the page to create a new user.

In the create user page, provide a username. Also, enable only programmatic access for the user, as we don't want the new user to login to the AWS portal.

The screenshot shows the 'Add user' page in the AWS IAM console. At the top, there are five numbered steps (1-5), with step 1 being the current step. The page is titled 'Set user details' and includes a sub-header: 'You can add multiple users at once with the same access type and permissions. Learn more'. There is a text input field for 'User name*' containing 's3user' and a button 'Add another user'. Below this is a section 'Select AWS access type' with the instruction: 'Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. Learn more'. Under 'Access type*', the 'Programmatic access' option is selected with a checkbox, with a description: 'Enables an access key ID and secret access key for the AWS API, CLI, SDK, and other development tools.' The 'AWS Management Console access' option is unselected, with a description: 'Enables a password that allows users to sign-in to the AWS Management Console.' At the bottom, there is a legend for '* Required', a 'Cancel' button, and a 'Next: Permissions' button.

Click on **Next:Permissions** to go to the next step.

Since the user is going to be reading and writing files via S3, let's give the user full access to S3. In the permissions page, Select **Attach existing policies** directly, and search for S3. You should see a policy named **AmazonS3FullAccess**. Select that option to allow the user complete access to S3. Once selected, click on **Next:Tags** to go to the next section.

Add user

1 2 3 4 5

Set permissions

Add user to group Copy permissions from existing user **Attach existing policies directly**

Create policy

Filter policies s3 Showing 4 results

	Policy name	Type	Used as
<input type="checkbox"/>	AmazonDMSRedshiftS3Role	AWS managed	None
<input checked="" type="checkbox"/>	AmazonS3FullAccess	AWS managed	None
<input type="checkbox"/>	AmazonS3ReadOnlyAccess	AWS managed	None
<input type="checkbox"/>	QuickSightAccessForS3StorageManagementAnalyticsReadOnly	AWS managed	None

Cancel Previous **Next: Tags**

Keep clicking on next and at the end you will see a **Create User**. Click on that to create the user. Once the user is created, AWS will display your Access Key and Secret on the page. Please copy them and make a note of them, as you will not be able to see them again.

Add user

1 2 3 4 5

Success
You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.
Users with AWS Management Console access can sign-in at: <https://319866474455.signin.aws.amazon.com/console>

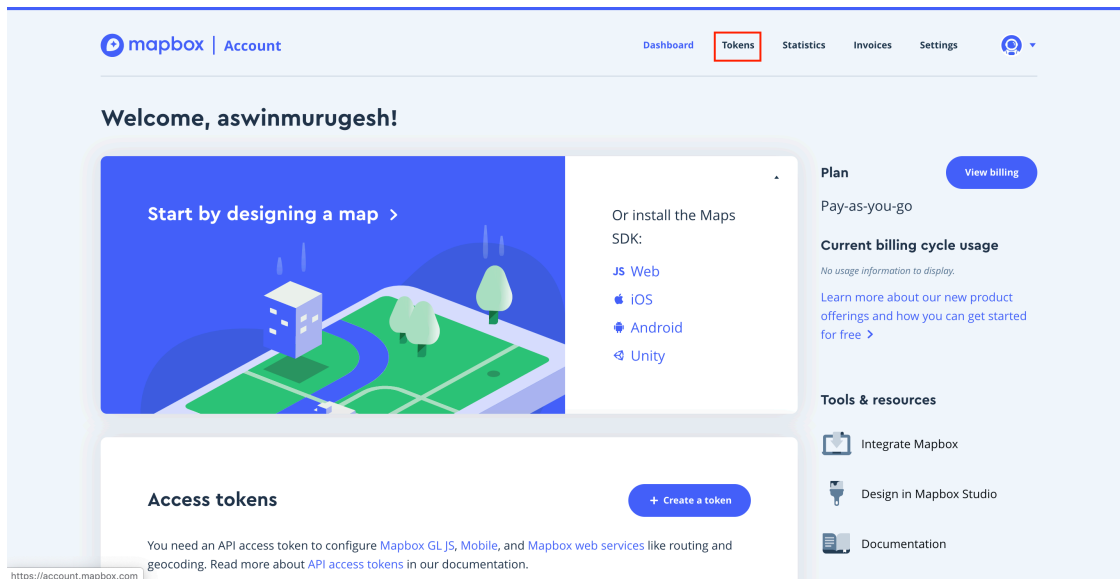
Download .csv

	User	Access key ID	Secret access key
<input checked="" type="checkbox"/>	s3User	AKIAJU6MHB7LWFF7CO2M	ottY1J2qVirkcGVKlzwBo8Rr7smFhIE8Wo1/qqq Hide

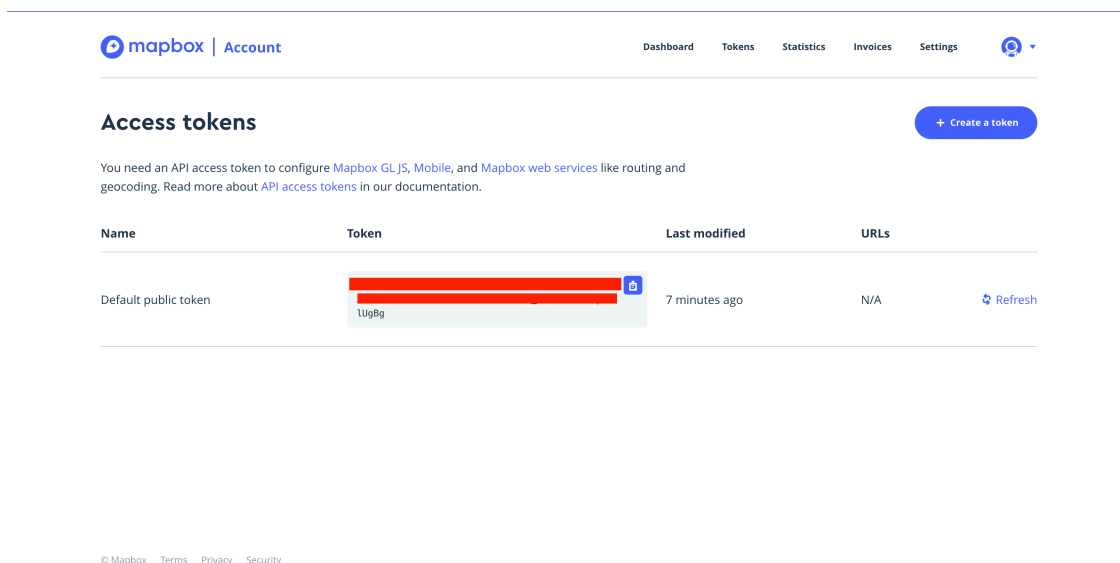
0.4 Setup Mapbox

Mapbox is a service that allows you to add interactive maps in your website. You can sign up to Mapbox in this link - <https://account.mapbox.com/auth/signup/>. Once you signup and login, you

should see the dashboard page as seen below. Click on the **Tokens** link on the top of the page



In the tokens list, you will see an existing token with the name **Default public token**; copy that token and add it to your code, to authenticate yourself.



0.5 Setting up Jupyter notebook in EC2

First, create a new EC2 instance (as taught in Week 3).

In the next couple of cases, you will be executing data wrangling operations in this newly created EC2 instance, using jupyter notebook. Let's setup the Jupyter notebook and dependencies for the case in this section.

1. Create virtualenv

```
sudo -H pip3 install --upgrade pip
sudo -H pip3 install virtualenv
virtualenv jupyter_env
```

2. Activate virtualenv

```
source jupyter_env/bin/activate
```

3. Install Jupyter inside the environment

```
pip install jupyter pandas numpy geopy boto3 shapely scikit-learn==0.21.3
```

4. Configure notebook to accept remote connections

```
jupyter notebook --generate-config
```

With the above step, there will be a configuration file generated in the `.jupyter` folder in your home folder. We need to update the config to allow remote connections. Next:

1. Add the below line at the end of the config file

```
c.NotebookApp.ip = '*'
```

2. Set a password for this Jupyter instance

```
jupyter notebook password
```

3. Start Jupyter

```
jupyter notebook --no-browser --port=8889
```

You can now access the remote instance via the URL `server_ip:8889`

0.6 Conclusion

1. We have now successfully uploaded the data files to S3 for case 5.1, and created an access key and secret that can be used to access the files from your Python code.
2. We have created and got access to a Mapbox account and an API key for the account that will be used in case 4.1 and 4.2
3. We have setup a new EC2 instance and configured a virtualenv so that we can do data wrangling operations on the full datasets in the cloud, in case 5.1 and 5.2