

# High Performance and Distributed Computing for Big Data

## Unit 3: Cloud Systems and Big Data Management

### Session 4 - AWS S3

---

Francesc Solsona Tehas [francesc.solsona@udl.cat](mailto:francesc.solsona@udl.cat)

Universitat Rovira i Virgili and Universitat de Lleida

- AWS S3: Storing files in the cloud
  - Installing AWS CLI
  - Configuring AWS credentials
  - Creating an S3 bucket
  - Syncing a local directory to upload to a bucket
  - Loading files from the bucket to the notebook from python
  - Writing files from the notebook to the bucket from python
  - Syncing a local directory to download from a bucket

## **S3 - Storing files in the cloud**

---

# What is S3?

Amazon S3 (Simple Storage Service) is an object storage service that offers industry-leading scalability, data availability, security, and performance. It is designed to store and retrieve any amount of data from anywhere on the web.



## S3 - Buckets and Objects

- **Buckets:** In Amazon S3, a bucket is a unique container for objects. Every object is stored within a bucket.
  - The bucket name must be globally unique across all existing bucket names in Amazon S3.
  - Buckets are used to store and organize objects.
- **Objects:** Objects are the fundamental entities within a bucket. They consist of data and metadata.
  - The information within an object is stored as a **key-value** pair.
  - The key, which is a unique identifier, is used to organize objects within the bucket. It is often formatted as a prefix to the object name.

For example, we can create a bucket called `hdcb-{your-name}` and store objects organized by session or other criteria.

```
data-{your-name }/  
  project1/data.csv  
  project2/data.csv
```

### Terms

- **Key** = *prefix* + *object name*
  - **prefix** = project1/ or project2/
  - **object name** = data.csv

Amazon S3 pricing is based on five factors:

1. **Storage Class:** The cost depends on the storage class used (Standard, Intelligent-Tiering, One Zone-IA, etc.).
2. **Storage:** The total volume of data stored per month.
3. **Requests:** The number and type of requests made.
4. **Data Transfer:** The cost of transferring data can vary by region and is also affected by whether data is transferred in or out.
5. **Management & Replication:** Additional features like data replication or management operations can also affect the cost.

For detailed information, you can refer to the [Amazon S3 Pricing Page](#).

## AWS CLI

---

## What is the AWS CLI?

The AWS CLI is a tool that allows you to interact with AWS services from the command line. It is a powerful tool that can be used to automate tasks and manage your AWS resources.

```
[ec2-user@ip-172-31-86-82 ~]$ aws

usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

    aws help
    aws <command> help
    aws <command> <subcommand> help

aws: error: the following arguments are required: command

[ec2-user@ip-172-31-86-82 ~]$
```

This is the base command which by itself doesn't do anything. What we are going to do now is to configure the AWS CLI with our credentials so we can later run commands that can access other AWS resources like S3.

# Installing the AWS CLI

Visit the [AWS CLI installation guide](#) to install the AWS CLI on your machine.

The screenshot shows the AWS CLI installation guide page. A red circle with the number '1' is placed over the 'macOS' section header. A red arrow points to a small circular icon in the bottom right corner of the terminal code block.

English ▾ Preferences ▾ Contact Us Feedback

Get started Service guides Developer tools AI resources

Search in this guide [Create an AWS Account](#)

This topic describes how to install or update the latest release of the AWS Command Line Interface (AWS CLI) on supported operating systems. For information on the latest releases of AWS CLI, see the [AWS CLI version 2 Changelog](#) on GitHub.

To install a past release of the AWS CLI, see [Installing past releases of the AWS CLI version 2](#). For uninstall instructions, see [Uninstalling the AWS CLI version 2](#).

**Important**  
AWS CLI versions 1 and 2 use the same `s3cli` command name. If you previously installed AWS CLI version 1, see [Migration guide for the AWS CLI version 2](#).

**Topics**

- [AWS CLI install and update instructions](#)
- [Troubleshooting AWS CLI install and uninstall errors](#)
- [Next steps](#)

**AWS CLI install and update instructions**

For installation instructions, expand the section for your operating system.

- [Linux](#)
- [macOS](#)
- [Windows](#)

**Install and update requirements**

- We support the AWS CLI on Microsoft-supported versions of 64-bit Windows.
- Admin rights to install software

**Install or update the AWS CLI**

To update your current installation of AWS CLI on Windows, download a new installer each time you update to overwrite previous versions. AWS CLI is updated regularly. To see when the latest version was released, see the [AWS CLI version 2 Changelog](#) on GitHub.

- Download and run the AWS CLI MSI installer for Windows (64-bit):  
<https://awscli.amazonaws.com/AWSCLIV2.msi>  
Alternatively, you can run the `msiexec` command to run the MSI installer.

```
C:\> msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi
```

For various parameters that can be used with `msiexec`, see [msiexec](#) on the Microsoft Docs website. For example, you can use the `/qn` flag for a silent installation.

**On this page**

- [AWS CLI install and update instructions](#)  
Troubleshooting AWS CLI install and uninstall errors  
Next steps

**Recently added to this guide**

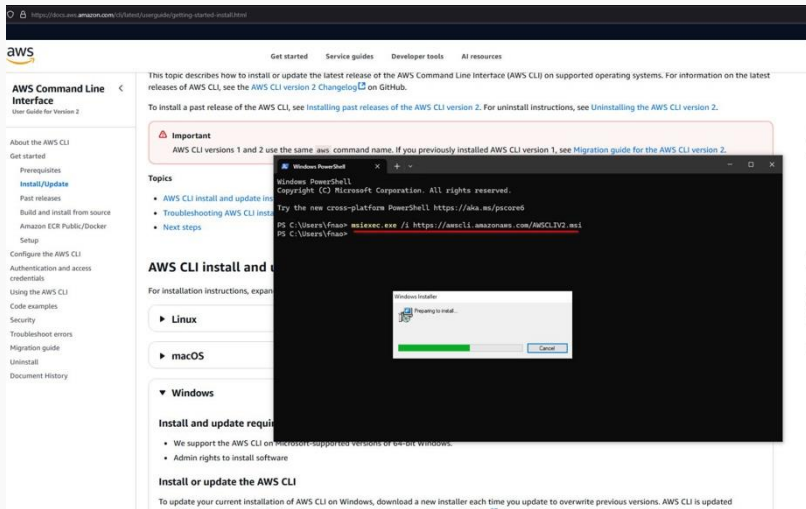
Did this page help you?

[Yes](#) [No](#)

[Provide feedback](#)

# Installing the AWS CLI

Paste the command on the terminal and wait for the installation to complete.



The screenshot shows the AWS Command Line Interface (CLI) installation page on the AWS website. The page title is "AWS Command Line Interface" and the URL is "https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html". The page content includes a navigation menu, a sidebar with "About the AWS CLI", "Get started", "Prerequisites", "Install/Update", "Past releases", "Build and install from source", "Amazon ECR Public/Docker", "Setup", "Configure the AWS CLI", "Authentication and access credentials", "Using the AWS CLI", "Code examples", "Security", "Troubleshoot errors", "Migration guide", "Uninstall", and "Document History". The main content area has an "Important" notice: "AWS CLI versions 1 and 2 use the same `aws` command name. If you previously installed AWS CLI version 1, see [Migration guide for the AWS CLI version 2](#)." Below this is a "Topics" section with links for "AWS CLI install and update instructions", "Troubleshooting AWS CLI installation", and "Next steps". The "AWS CLI install and update instructions" section is expanded, showing "Linux", "macOS", and "Windows" options. The "Windows" section is further expanded, showing "Install and update requirements" (supporting 64-bit Windows and requiring admin rights) and "Install or update the AWS CLI" (instructions to download a new installer to overwrite previous versions). A Windows PowerShell terminal window is overlaid on the page, showing the command `PS C:\Users\fnao> msisexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi`. A Windows Installer dialog box is also visible, showing "Preparing to install..." with a progress bar and a "Cancel" button.

# Installing the AWS CLI

Or if you are on MacOS, go to the MacOS section and also paste the corresponding commands on your terminal.

**macOS**

### Install and update requirements

- We support the AWS CLI on macOS versions 11 and later. For more information, see [macOS support policy updates for the AWS CLI v2](#) on the AWS Developer Tools Blog.
- Because AWS doesn't maintain third-party repositories, we can't guarantee that they contain the latest version of the AWS CLI.

**macOS version support matrix**

AWS CLI version	Supported macOS version
2.21.0 – current	11+
2.17.0 – 2.20.0	10.15+
2.0.0 – 2.16.12	10.14 and below

### Install or update the AWS CLI

If you are updating to the latest version, use the same installation method that you used in your current version. You can install the AWS CLI on macOS in the following ways.

GUI installer   **Command line installer - All users**   Command line - Current user

If you have `sudo` permissions, you can install the AWS CLI for all users on the computer. We provide the steps in one easy to copy and paste group. See the descriptions of each line in the following steps.

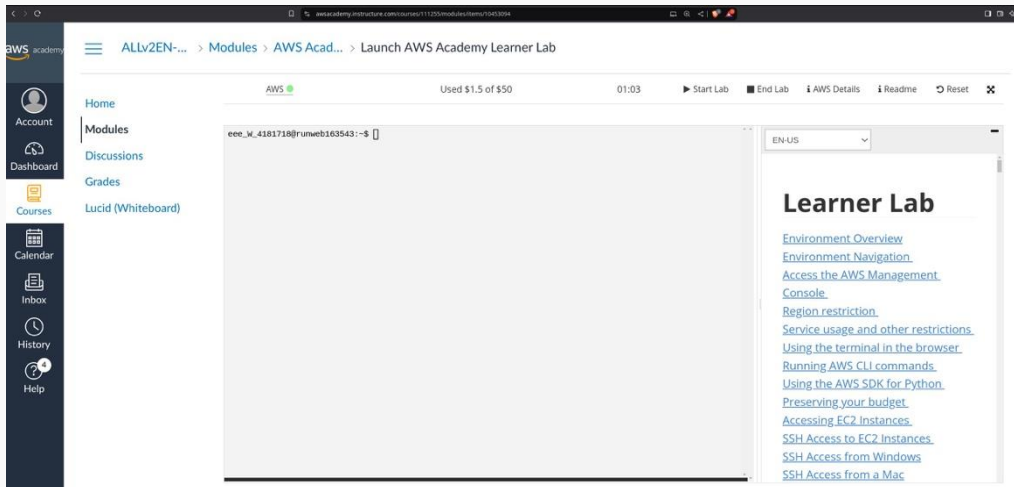
```
$ curl "https://awscli.amazonaws.com/AWSCLIV2.pkg" -o "AWSCLIV2.pkg"
$ sudo installer -pkg AWSCLIV2.pkg -target /
```

### Guided installation instructions

- Download the file using the `curl` command. The `-o` option specifies the file name that the downloaded package is written to. In this example, the file is written to `AWSCLIV2.pkg` in the current folder.

# Configuring the AWS Credentials

We're now going to visit the Learner Lab page on the AWS Academy website to get our credentials. You have [this guide](#) and [this guide](#) available on the subject's [website](#) to help you with setting up AWS. Wait until the lab loads and you see the page below.



The screenshot shows a web browser window displaying the AWS Academy Learner Lab interface. The browser's address bar shows the URL `awsacademy.structure.com/courses/11125/modules/items/1003894`. The page title is "Launch AWS Academy Learner Lab".

The interface includes a dark sidebar on the left with navigation options: Account, Dashboard, Courses, Calendar, Inbox, History, and Help. The main content area has a breadcrumb trail: "ALLv2EN-... > Modules > AWS Acad... > Launch AWS Academy Learner Lab".

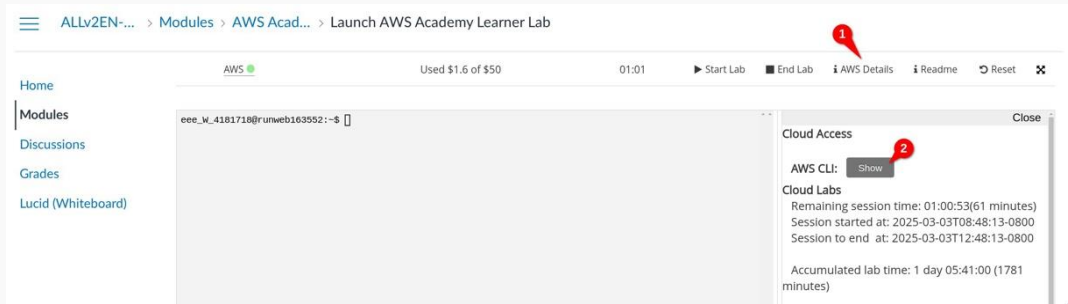
At the top of the main content area, there is a status bar showing "AWS" with a green dot, "Used \$1.5 of \$50", and a timer "01:03". Action buttons include "Start Lab", "End Lab", "AWS Details", "Readme", and "Reset".

The central part of the interface features a terminal window with the prompt `eee_w_4181718@runweb163543:~$`. To the right of the terminal is a "Learner Lab" panel with a dropdown menu set to "EN-US". This panel contains a list of links for navigation:

- [Environment Overview](#)
- [Environment Navigation](#)
- [Access the AWS Management Console](#)
- [Region restriction](#)
- [Service usage and other restrictions](#)
- [Using the terminal in the browser](#)
- [Running AWS CLI commands](#)
- [Using the AWS SDK for Python](#)
- [Preserving your budget](#)
- [Accessing EC2 Instances](#)
- [SSH Access to EC2 Instances](#)
- [SSH Access from Windows](#)
- [SSH Access from a Mac](#)

# Configuring the AWS Credentials

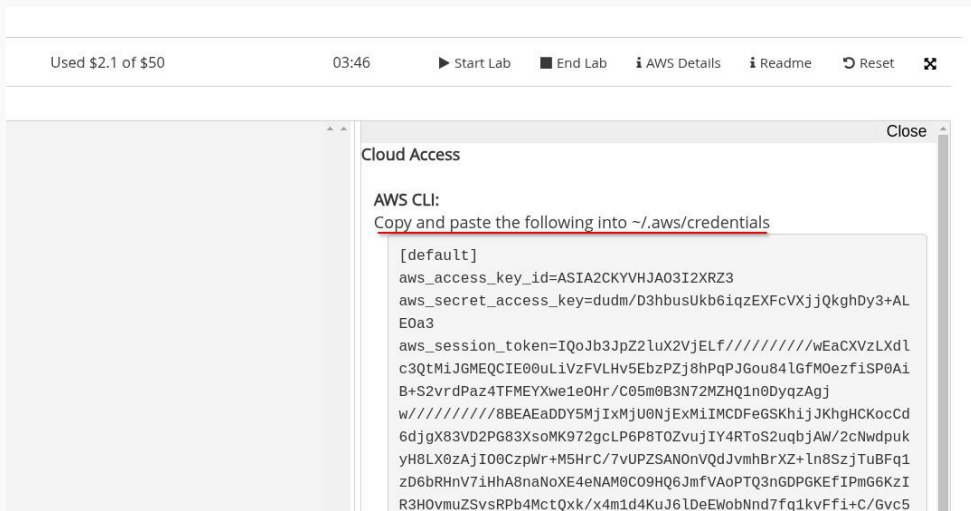
Now click on `AWS Details` and then on `Show` to reveal your credentials.



The screenshot displays the AWS Academy Learner Lab interface. The breadcrumb navigation at the top reads: `ALLv2EN-... > Modules > AWS Acad... > Launch AWS Academy Learner Lab`. The main header area includes the text `AWS` with a green status indicator, `Used $1.6 of $50`, a timer at `01:01`, and buttons for `Start Lab`, `End Lab`, `AWS Details`, `Readme`, and `Reset`. A red callout bubble with the number `1` points to the `AWS Details` button. On the left sidebar, the `Modules` section is expanded, showing `Home`, `Modules`, `Discussions`, `Grades`, and `Lucid (Whiteboard)`. The central terminal window shows the prompt `eee_w_4181718@runweb163552:~$`. On the right, the `Cloud Access` panel is open, featuring a `Show` button for `AWS CLI:` with a red callout bubble containing the number `2`. Below this, the `Cloud Labs` section displays session information: `Remaining session time: 01:00:53(61 minutes)`, `Session started at: 2025-03-03T08:48:13-0800`, `Session to end at: 2025-03-03T12:48:13-0800`, and `Accumulated lab time: 1 day 05:41:00 (1781 minutes)`.

# Configuring the AWS Credentials

You'll see some text containing your credentials.



Used \$2.1 of \$50      03:46      ▶ Start Lab      ■ End Lab      ⓘ AWS Details      ⓘ Readme      ↻ Reset      ✕

Cloud Access Close

**AWS CLI:**  
Copy and paste the following into ~/.aws/credentials

```
[default]
aws_access_key_id=ASIA2CKYVHJA03I2XRZ3
aws_secret_access_key=dudm/D3hbusUkb6iqzEXFcVXjjQkghDy3+AL
EOa3
aws_session_token=IQoJb3JpZ2luX2VjELf////////wEaCXVzLXd1
c3QtMiJGMEQCIE00uLiVzFVLHv5EbzPZj8hPqPJGou84lGfM0ezfiSP0Ai
B+S2vrdPaz4TFMEYXwe1e0Hr/C05m0B3N72MZHQ1n0DyqzAgj
w////////8BEAEaDDY5MjIxMjU0NjExMiIMCDFeGSKhijJKhgHCKocCd
6djgX83VD2PG83XsoMK972gcLP6P8T0ZvujiY4RToS2uqbJAW/2cNwdpuk
yH8LX0zAjI00Czpwr+M5HrC/7vUPZSANonVQdJvvhBrXZ+ln8SszjTuBFq1
zD6bRHnV7iHhA8naNoXE4eNAM0C09HQ6JmfVAoPTQ3nGDPGKEfIPmG6KzI
R3HOvmuZSvsRPb4MctQxk/x4m1d4KuJ6lDeEWobNnd7fq1kvFfi+C/Gvc5
```

## Configuring the AWS Credentials

Next we need to paste that text to a file called `credentials` inside the `.aws` folder in our home directory.

Make sure the `.aws` folder exists on your local machine by running `mkdir .aws` command (remember if it throws an error there's nothing to worry about, it just means the folder already exists). Now we are going to create the `credentials` file inside the `.aws` folder. Run the following command:

```
notepad .aws/credentials.
```

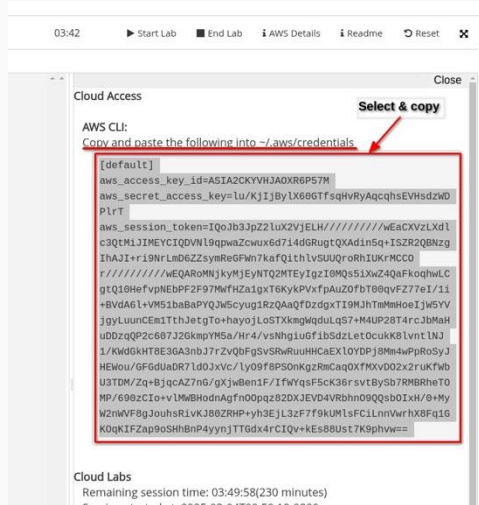
or for MacOS users:

```
open .aws/credentials.
```

This will open a text editor where you can write the credentials.

# Configuring the AWS Credentials

Go back to the AWS Academy website and copy the text containing your credentials.



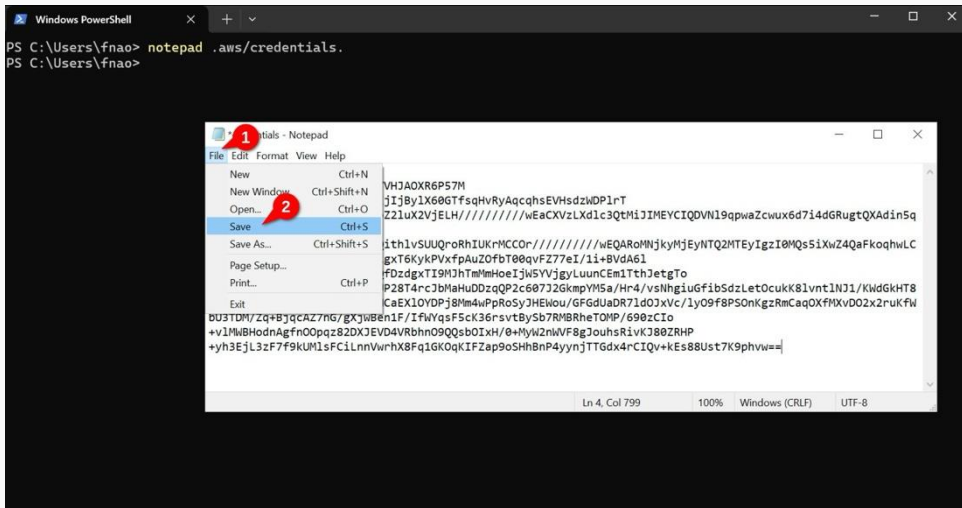
The screenshot shows a web interface for an AWS Academy lab. At the top, there is a navigation bar with buttons for 'Start Lab', 'End Lab', 'AWS Details', 'Readme', 'Reset', and a close icon. Below this, the main content area is titled 'Cloud Access'. Underneath, it says 'AWS CLI:' followed by the instruction 'Copy and paste the following into ~/.aws/credentials'. A red box highlights the following text, and a red arrow points to it with the label 'Select & copy'.

```
[default]
aws_access_key_id=ASIA2CKYVH3A0XR6P57M
aws_secret_access_key=lu/KjIjByLx60GTfsqHvRyAqcqhsEVHsdzWD
PlrT
aws_session_token=IQoJb3JpZ2luX2VjELH////////wEaCXVzLXdI
c3QtMjIIMEYCIQDVNl9qpwaZcwux6d7i4dGRugtQXAdin5q+ISZR2QBnzg
IhAJI+r19NrLmd6ZZsymReGFwn7kaFqithlvsUUQroRhIUkrMCCO
r////////wEQARoMnjkyMjEyNTQ2MTEyIgzI0MQs5iXwZ4QaFkoqhwLC
gtQ10HefvpNEbPF2F97MwFhZa1gxT6KykPVxfpAuZ0fbT00qvFZ77eI/i
+BvDA6l+VM51baBaPYQJw5cyug1RzQAaQfDzdgxTI9MJhTmMmHoeIjw5YV
jgyLuunCEm1TthJEtgTo+hayojLoSTXkmgWqduLqS7+M4UP28T4rcJbMaH
uDDzqQP2c687J2GkmpYM5a/Hr4/vsNhg1uGf1bSdzLetOcukK8lvntlNJ
1/KwDgkHTBE3GA3nbJ7rZvQbFgSvSRwRuuHHCaEXlOYDPj8m4wPpRoSyJ
HEWou/6FGdUaDR7ld0JxVc/ly09f8P50nKgZrMcaQ0XfMxvD02x2ruKfWb
U3TDM/Zq+BjqcAZ7n6/gXjwBen1F/IfWYqsF5cK36rsvtBySb7RMBRheTO
MP/690zCIo+vLMWBhodnAgfn00pqz82DXJEVD4VRbhn09QQsb0IxH/0+My
W2nWVfBgJouhsRlVkJB0ZRHP+yh3EjL3zF7f9KUMlsFC1LnnVvrhX8Fq1G
K0qKIFZap9oSshBnP4yynjTTGdx4rCIQv+kEs88Ust7K9phvw==
```

Below the credentials, there is a 'Cloud Labs' section with the text 'Remaining session time: 03:49:58(230 minutes)'.

## Configuring the AWS Credentials

Now go back to the text editor. Paste the credentials and save the file. You can now exit the text editor.



The screenshot shows a Windows PowerShell terminal window and a Notepad text editor window. The PowerShell window shows the command `notepad .aws/credentials.` being executed. The Notepad window shows the contents of the `.aws/credentials` file, which contains the AWS Access Key ID and Secret Access Key. A red circle with the number 1 is placed over the Notepad window title bar, and a red circle with the number 2 is placed over the 'Save' option in the File menu.

```
PS C:\Users\fnao> notepad .aws/credentials.  
PS C:\Users\fnao>
```

```
File Edit Format View Help  
New Ctrl+N  
New Window Ctrl+Shift+N  
Open... Ctrl+O  
Save Ctrl+S  
Save As... Ctrl+Shift+S  
Page Setup...  
Print... Ctrl+P  
Exit  
VHJAOXR6P57M  
jIjBylX60GTfsqHvRyAqcqhsEVHsdzWDP1rT  
Z2luX2VjELH////////wEaCXvzLXd1c3QtMiJIMEYCIQDVN19qpwaZcwux6d7i4dGRugtQXAdin5q  
ith1vSUUQroRhIUkrMCCOr////////wEQARoMNjkyMjEyNTQ2HTEyIgzI0MQs5iXwZ4QaFkoqhwLC  
gxT6KykPVxfpAuZOfbT00qvFZ77eI/i+BVdA61  
fDzdgxTI9MJhTmMmHoeIjw5YVjgyLuunCEm1TthJtgTo  
P28T4rcJbMaHuDDzqQP2c607J2GkmpYM5a/Hr4/vsNhgIuGfibSdzLetOocukK8lvnt1NJ1/KwdGkHT8  
CaEX1OYDPj8Mm4wPpRoSyJHEWou/GFGdUaDR71d0JxVc/lyO9f8PSONkgzRmCaqOXfMXvD02x2ruKfW  
DU3TDM/Zq+BjJqCAZ7HG/gXjWBen1F/IfwYqsF5cK36rsvtBySb7RMBRheTOMP/690zCIo  
+v1MWBHodnAgfn0Opqz82DXJEVD4VRbhn09QQsb0IxH/0+MyW2nWVF8gJouhsRivKJ80ZRHP  
+yh3EjL3zF7f9KUM1sFc1LnnVwrhX8Fq1GKOqKIFZap9oSHhBnP4yynjTTGdx4rCIQv+kEs88Ust7K9phvw==
```

## Configuring the AWS Credentials

We can check the contents of the file using `cat`:

```
PS C:\Users\fnao> cat .aws\credentials
[default]
aws_access_key_id=ASIA2CKYVHJAOXR6P57M
aws_secret_access_key=lu/KjIjBylX60GTfsqHvRyAqcqhsEVHsdzWDPlrT
aws_session_token=IQoJb3JpZ2...
PS C:\Users\fnao>
```

## Configuring the AWS Credentials

To test if the configuration was successful, run `aws sts get-caller-identity` and you should see something like this:

```
PS C:\Users\fnao> aws sts get-caller-identity
{
  "UserId": "AROAI2CKYVHJALK46ZMHVM:user3869188=Ferran_Aran_Test",
  "Account": "692212546112",
  "Arn": "arn:aws:sts::692212546112:assumed-role/voclabs/user3869188=Ferran_Aran_Test"
}

PS C:\Users\fnao>
```

Great! We can now run AWS CLI commands on our local machine to manage our AWS resources. For example, we can upload files to an S3 bucket.

## S3 Buckets

---

# Creating a S3 bucket

Use the searchbar to head to the S3 service.



# Creating a S3 bucket

Click on `Create bucket`.

The screenshot shows the Amazon S3 console interface. At the top, there is a navigation bar with the AWS logo, a search bar, and various utility icons. Below this, the main content area is titled 'Amazon S3' and features a left-hand navigation menu with options like 'General purpose buckets', 'Directory buckets', and 'Storage Lens'. The main panel displays an 'Account snapshot' section and a 'General purpose buckets' section. In the 'General purpose buckets' section, there is a search bar and a table of buckets. The 'Create bucket' button is highlighted with a red circle and the number 1.

**Account snapshot - updated every 24 hours** All AWS Regions [View Storage Lens dashboard](#)  
Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. [Learn more](#)

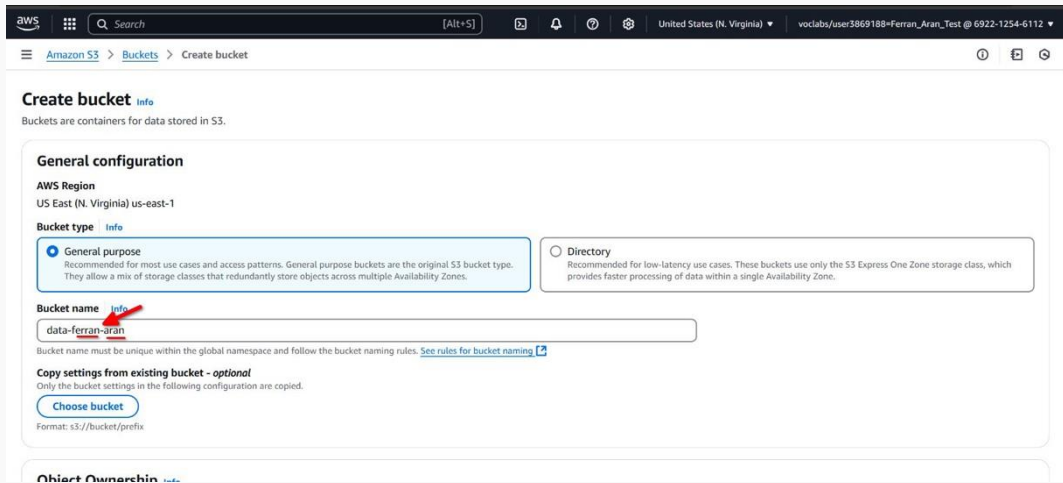
**General purpose buckets** All AWS Regions [Copy ARN](#) [Empty](#) [Delete](#) **Create bucket**

Buckets are containers for data stored in S3.

Name	AWS Region	IAM Access Analyzer
<a href="#">ferran-test123</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>

# Creating a S3 bucket

Name the bucket `data-{your-name}`. For example I will be naming it `data-ferran-aran`. **Bucket names have to be unique across all of AWS.**



aws [Search] [Alt+S] United States (N. Virginia) ▼ voclabs/user3869188+Ferran\_Aran\_Test @ 6922-1254-6112 ▼

Amazon S3 > Buckets > Create bucket

## Create bucket [info](#)

Buckets are containers for data stored in S3.

### General configuration

**AWS Region**  
US East (N. Virginia) us-east-1

**Bucket type** [Info](#)

- General purpose**  
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.
- Directory**  
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

**Bucket name** [Info](#)

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

**Copy settings from existing bucket - optional**  
Only the bucket settings in the following configuration are copied.

Format: s3://bucket/prefix

### Object Ownership [info](#)

# Creating a S3 bucket

Leave everything else as default, scroll all the way down and click on `Create bucket`.

Amazon S3 > Buckets > Create bucket

[Add tag](#)

**Default encryption** [Info](#)  
Server-side encryption is automatically applied to new objects stored in this bucket.

**Encryption type** [Info](#)

- Server-side encryption with Amazon S3 managed keys (SSE-S3)
- Server-side encryption with AWS Key Management Service keys (SSE-KMS)
- Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)  
Secure your objects with two separate layers of encryption. For details on pricing, see DSSE-KMS pricing on the Storage tab of the [Amazon S3 pricing page](#).


**Bucket Key**  
Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. [Learn more](#)

- Disable
- Enable

▶ **Advanced settings**

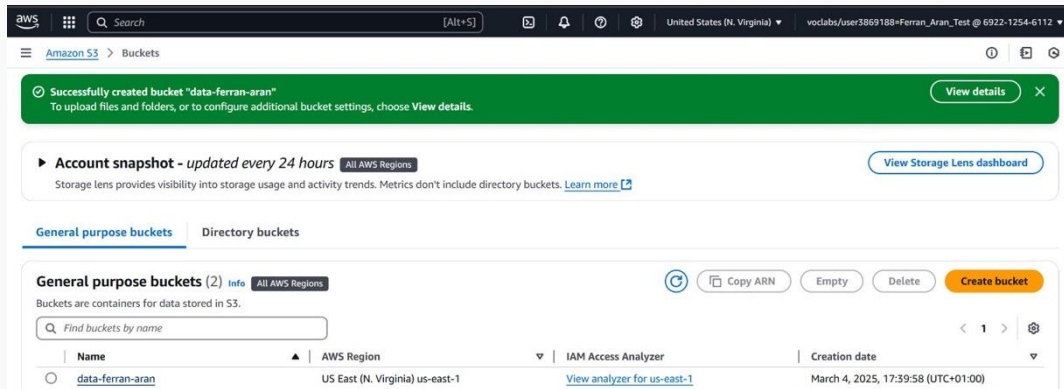
[i](#) After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.

[Cancel](#) [Create bucket](#)



# Creating a S3 bucket

You will now be headed to the S3 dashboard. You should see a success message and the bucket you just created.



The screenshot shows the AWS S3 console interface. At the top, there's a navigation bar with the AWS logo, a search bar, and user information. Below that, a green banner displays a success message: "Successfully created bucket 'data-ferran-aran'". Below the banner, there's a section for "Account snapshot" and a "View Storage Lens dashboard" button. The main content area is titled "General purpose buckets" and shows a list of buckets. The first bucket listed is "data-ferran-aran" in the "us-east-1" region, created on March 4, 2025.

Amazon S3 > Buckets

Successfully created bucket "data-ferran-aran"  
To upload files and folders, or to configure additional bucket settings, choose [View details](#).

Account snapshot - updated every 24 hours [All AWS Regions](#) [View Storage Lens dashboard](#)

Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. [Learn more](#)

General purpose buckets | Directory buckets

General purpose buckets (2) [Info](#) [All AWS Regions](#)

Buckets are containers for data stored in S3.

Find buckets by name

Name	AWS Region	IAM Access Analyzer	Creation date
<a href="#">data-ferran-aran</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	March 4, 2025, 17:39:58 (UTC+01:00)

# Uploading a file to the bucket

## From the S3 dashboard

1. Click on the bucket.
2. Create a folder and name it *test*.
3. Click on the folder.
4. Click on Upload.
5. Click on Add files and select an example file.
6. Click Upload.

Our bucket **data-{your-name}/** now contains one file. It is stored in **s3://data-{your-name}/test**. This file is private by default. Only the owner can access them.

## From the AWS CLI

- `aws s3 cp <local-file-path> s3://<bucket-name>/<optional-target-name>`

**Example (upload photo.jpg to a bucket named my-bucket):**

```
aws s3 cp photo.jpg s3://my-bucket/
```

## Sync with a bucket

---

# Syncing a local directory to upload to a bucket

## From the AWS CLI

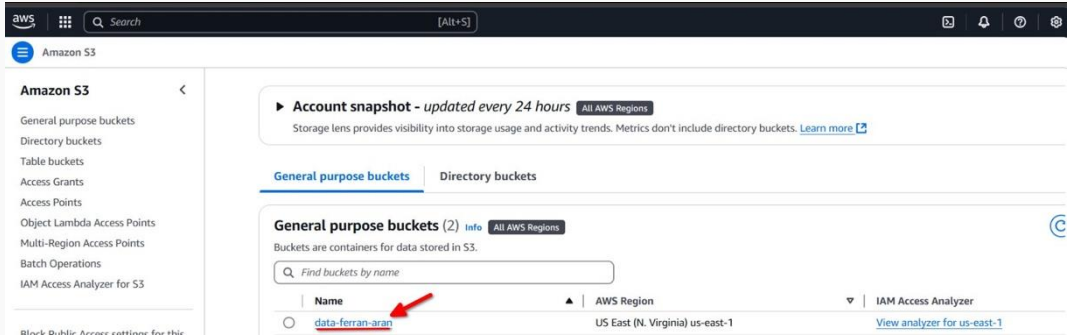
1. Go to your desktop and create a folder named `data`.
2. Create a file named `my-dataset.txt` and write some text in it. Save it.
3. Open a terminal and navigate to the Desktop directory. Use `cd Desktop`.
4. Run `aws s3 sync data s3://data-{your-name}` to upload the files to the bucket. In my case this is the result:

```
PS C:\Users\fnao\Desktop> aws s3 sync data s3://data-ferran-aran  
upload: data\my-dataset.txt to s3://data-ferran-aran/my-dataset.txt
```

That's it! You have now uploaded a file to the bucket!

# Inspecting the bucket from the AWS Console

Go to the S3 dashboard on AWS and click on the bucket we just created.



The screenshot shows the AWS S3 console interface. At the top, there's a search bar and navigation icons. The left sidebar lists various S3 features, with 'Amazon S3' selected. The main content area shows an 'Account snapshot' section and a 'General purpose buckets' section. The 'General purpose buckets' section contains a table of buckets. A red arrow points to the bucket named 'data-ferran-aran'.

**Account snapshot - updated every 24 hours** All AWS Regions  
Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. [Learn more](#)

**General purpose buckets** | Directory buckets

**General purpose buckets (2)** info All AWS Regions

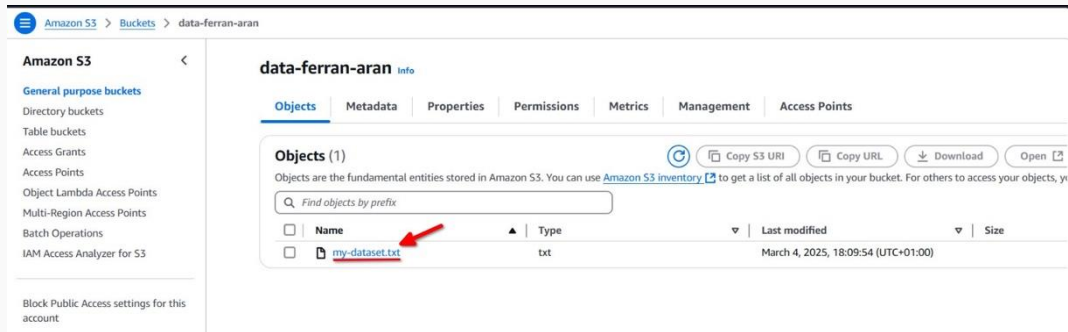
Buckets are containers for data stored in S3.

Find buckets by name

Name	AWS Region	IAM Access Analyzer
<a href="#">data-ferran-aran</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>

# Inspecting the bucket from the AWS Console

You will see the contents of the bucket, in this case the file `my-dataset.txt`.



The screenshot shows the AWS S3 console interface for the bucket 'data-ferran-aran'. The left sidebar lists various S3 features, and the main content area displays the bucket's contents. A table lists one object, 'my-dataset.txt', which is highlighted with a red arrow. The table columns are Name, Type, Last modified, and Size.

Amazon S3 > Buckets > data-ferran-aran

### Amazon S3

- General purpose buckets
- Directory buckets
- Table buckets
- Access Grants
- Access Points
- Object Lambda Access Points
- Multi-Region Access Points
- Batch Operations
- IAM Access Analyzer for S3

Block Public Access settings for this account

## data-ferran-aran

Info

- Objects
- Metadata
- Properties
- Permissions
- Metrics
- Management
- Access Points

Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you can use [Access Points](#).

Find objects by prefix

<input type="checkbox"/>	Name	Type	Last modified	Size
<input type="checkbox"/>	<a href="#">my-dataset.txt</a>	txt	March 4, 2025, 18:09:54 (UTC+01:00)	

# Inspecting the bucket from the AWS Console

Click on the file to see further details.

The screenshot shows the AWS S3 console interface. The breadcrumb navigation at the top reads: Amazon S3 > Buckets > data-ferran-aran > my-dataset.txt. The main content area is titled 'my-dataset.txt' and includes tabs for 'Properties', 'Permissions', and 'Versions'. The 'Properties' tab is active, displaying an 'Object overview' section with the following details:

- Owner:** awslabsc0w490775111669516777
- AWS Region:** US East (N. Virginia) us-east-1
- Last modified:** March 4, 2025, 18:09:54 (UTC+01:00)
- Size:** 18.0 B
- Type:** txt
- Key:** my-dataset.txt

On the right side of the 'Object overview' section, there are three metadata fields:

- S3 URI:** <s3://data-ferran-aran/my-dataset.txt> (highlighted with a red arrow)
- Amazon Resource Name (ARN):** <arn:aws:s3:::data-ferran-aran/my-dataset.txt>
- Entity tag (Etag):** [Ofee9acadfe8753f721d44b985a90c67](#)

Below these fields is the **Object URL:** <https://data-ferran-aran.s3.us-east-1.amazonaws.com/my-dataset.txt>

At the top right of the object details area, there are four buttons: 'Copy S3 URI', 'Download', 'Open', and 'Object actions'.

The left sidebar shows the 'Amazon S3' navigation menu with categories like 'General purpose buckets', 'Storage Lens', and 'Dashboards'.

## Working with S3 from python

---

## Configuring AWS credentials on an EC2 instance

The first thing we'll have to do is to connect to our EC2 instance we configured during last session. More information on last session can be found [here \(Session 3\)](#).

Once we are connected through SSH on a remote terminal, we'll need to configure AWS Credentials similarly to how we did on our local machine. But this time we will have to use a terminal editor.

If this steps get confusing I suggest checking [this guide](#) on the subject's website for a more detailed explanation.

## Configuring AWS credentials on an EC2 instance

EC2 machines come with AWS CLI already installed so we won't have to worry about that.

Remember we first need to make sure the `.aws` folder exists in the home directory of the user we are using. If it doesn't exist we can create it by running `mkdir .aws`.

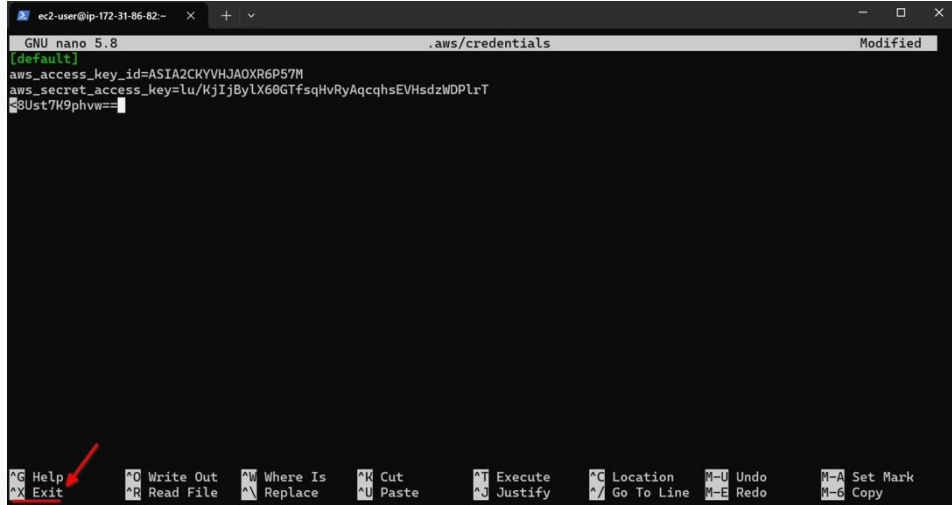
Next we need to create the `credentials` file inside the `.aws` folder. We can do this by running `nano .aws/credentials`.

The nano text editor will open and we can paste the credentials we copied from the AWS Academy website.

To save the file we can press `Ctrl + X` and then `Y` and finally `Enter`. See the following screenshots of the process.



# Configuring AWS credentials on an EC2 instance



The screenshot shows a terminal window with a nano editor open. The editor is editing a file named `.aws/credentials`. The content of the file is as follows:

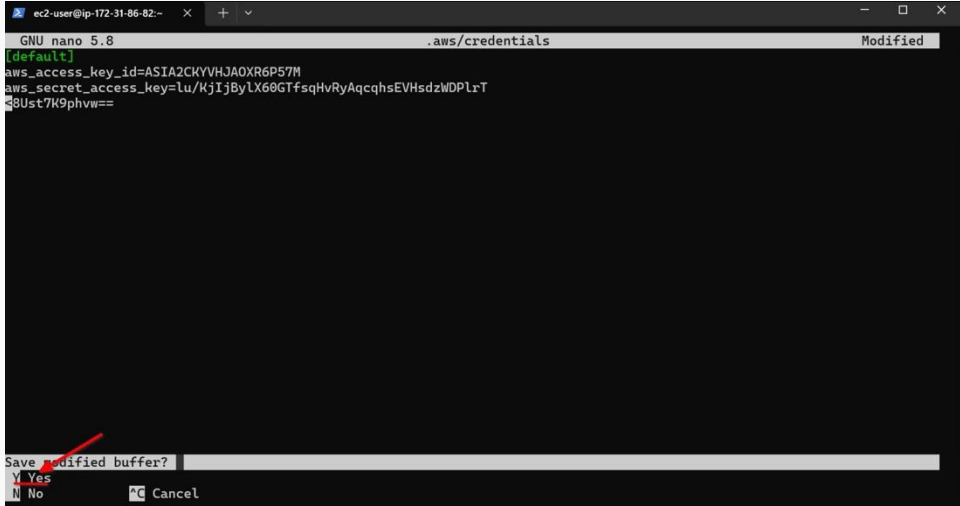
```
GNU nano 5.8 .aws/credentials Modified
[default]
aws_access_key_id=ASIA2CKYVHJA0XR6P57M
aws_secret_access_key=Lu/KjIjByLX60GTfsqHvRyAqcqhsEVHsdzWDPLrT
8Ust7K9phvw==
```

At the bottom of the terminal, there is a help menu with the following options:

<b>^G</b> Help	<b>^O</b> Write Out	<b>^W</b> Where Is	<b>^K</b> Cut	<b>^T</b> Execute	<b>^C</b> Location	<b>M-U</b> Undo	<b>M-A</b> Set Mark
<b>^X</b> Exit	<b>^R</b> Read File	<b>^_\</b> Replace	<b>^U</b> Paste	<b>^J</b> Justify	<b>^/</b> Go To Line	<b>M-E</b> Redo	<b>M-6</b> Copy

A red arrow points to the `^X` Exit option in the help menu.

# Configuring AWS credentials on an EC2 instance

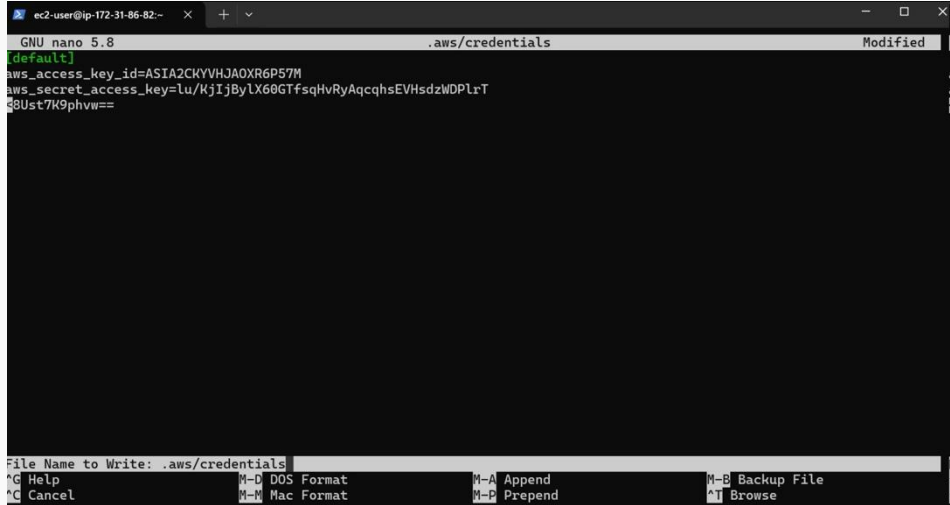


The screenshot shows a terminal window with the title bar "ec2-user@ip-172-31-86-82:~". The terminal content is as follows:

```
GNU nano 5.8 .aws/credentials Modified
[default]
aws_access_key_id=ASIA2CKYVHJAOXR6P57M
aws_secret_access_key=lu/KjIjBylX60GTfsqHvRyAqcqhsEVHsdzWDPlrT
8Ust7K9phvw==
```

At the bottom of the terminal, a prompt asks "Save modified buffer?". The letter "Y" is highlighted with a red circle, and a red arrow points to it from the left. Below the prompt, the options "Y Yes" and "N No" are visible, along with a "Ctrl C Cancel" option.

# Configuring AWS credentials on an EC2 instance



The image shows a terminal window with a nano editor editing a file named `.aws/credentials`. The terminal title bar indicates the user is `ec2-user` on an instance with IP `172-31-86-82`. The editor shows the following content:

```
GNU nano 5.8 .aws/credentials Modified
[default]
aws_access_key_id=ASIA2CKYVHJA0XR6P57M
aws_secret_access_key=lu/kjIjBylX60GTfsqHvRyAqcqhsEVHsdzWDPLrT
8Ust7K9phvw==
```

The bottom of the terminal shows the nano editor's status bar with the file name and various keyboard shortcuts:

```
File Name to Write: .aws/credentials
^G Help          M-D DOS Format  M-A Append      M-B Backup File
^C Cancel        M-M Mac Format   M-P Prepend     ^T Browse
```

## Configuring AWS credentials on an EC2 instance

We can now test if the configuration was successful by running `aws sts get-caller-identity`.

If the configuration was successful we should see something like this:

```
{
  "UserId": "AROIA2CKYVHJALK46ZMHVM:user3869188=Ferran_Aran_Test",
  "Account": "692212546112",
  "Arn": "arn:aws:sts::692212546112:assumed-role/voclabs/user3869188=Ferran_Aran_Test"
}
```

## Reading files from S3 with python

We are going to use the *boto3 python library* to access and write S3 files from a jupyter notebook. This library allows Python developers to write software that makes use of services like Amazon S3 on AWS.

We'll need to first activate one of the environment we created during the last session. For example, I will be using `project1` environment. Follow the steps below to activate the environment and install `boto3`:

```
cd project1
source .project1/bin/activate
pip install boto3
```

Once that is done, launch the jupyter server with the command below:

```
jupyter notebook --no-browser --port=8888 --ip=0.0.0.0
```

Remember on last session we saw the steps to access the jupyter notebook from our local machine. If you need a refresher you can check [Session 3](#) on the subject's website.

## Reading files from S3 with python

Open a jupyter notebook and paste the following code:

```
import boto3

DATA_BUCKET_NAME = "data-your-name"
DATA_FILE_NAME = "my-dataset.txt" # Path to the file in S3

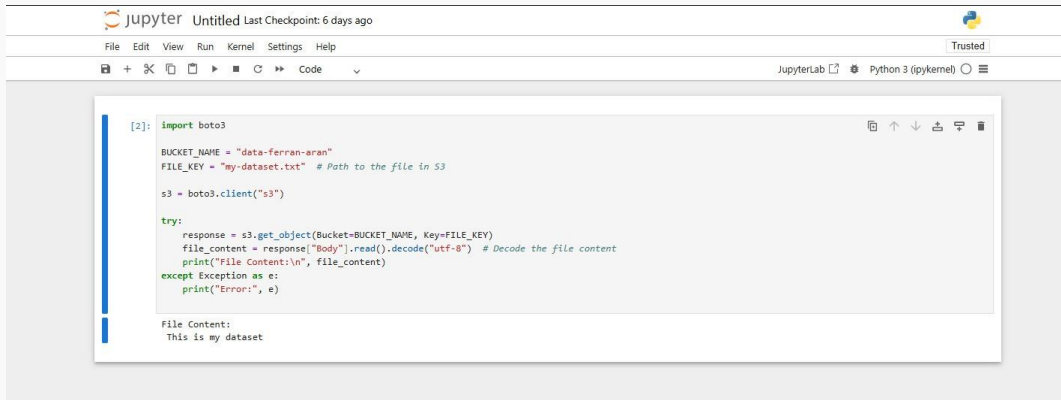
s3 = boto3.client("s3")

response = s3.get_object(Bucket=DATA_BUCKET_NAME, Key=DATA_FILE_NAME)
file_content = response["Body"].read().decode("utf-8") # Decode the file content
print("File Content:\n", file_content)
```

Be careful to replace `data-your-name` with the name of the bucket you created and `my-dataset.txt` with the name of the file you uploaded.

# Reading files from S3 with python

If everything is correct you should see the content of the file printed in the notebook.



The image shows a JupyterLab interface with a code cell containing Python code to read a file from Amazon S3. The code uses the boto3 library to create an S3 client, retrieve an object from a bucket, and print its content. The output of the code cell shows the file content: "This is my dataset".

```
[2]: import boto3

BUCKET_NAME = "data-ferran-aran"
FILE_KEY = "my-dataset.txt" # Path to the file in S3

s3 = boto3.client("s3")

try:
    response = s3.get_object(Bucket=BUCKET_NAME, Key=FILE_KEY)
    file_content = response["Body"].read().decode("utf-8") # Decode the file content
    print("File Content:\n", file_content)
except Exception as e:
    print("Error:", e)
```

File Content:  
This is my dataset

## RECAP: Accessing the file from the notebook

To access the file from the notebook we have used our aws-credentials for the current user (owner of the bucket).

### Be aware!

Someone could log into your jupyter instance in the browser and access the file using your credentials.

### What to do?

In real life, we would use IAM roles to give the notebook the necessary permissions to access the file.

**But, in AWS Educate, we can not use IAM roles.**

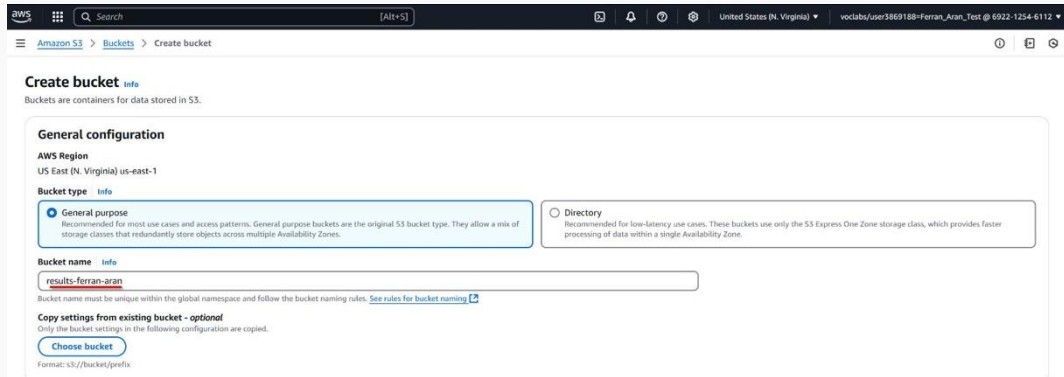
### Another option

Make the file public and access it without credentials :)

# Writing files to S3 with python

We can also write files to S3 using boto3. Lets first create another bucket to store the files we will write from the notebook. This one we will call `results-{your-name}`.

Leave everything as default like before and scroll all the way down to click on `Create bucket`.



The screenshot shows the AWS console interface for creating a new bucket. At the top, the navigation bar includes the AWS logo, a search bar, and the user's profile information. The breadcrumb trail indicates the current location: Amazon S3 > Buckets > Create bucket. The main heading is "Create bucket" with an "Info" link. Below this, a brief description states "Buckets are containers for data stored in S3." The "General configuration" section is expanded, showing the "AWS Region" set to "US East (N. Virginia) us-east-1". Under "Bucket type", two options are visible: "General purpose" (selected with a radio button) and "Directory". The "General purpose" option is highlighted with a blue border and contains the text: "Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones." The "Directory" option is unselected and contains the text: "Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone." Below the bucket type selection, the "Bucket name" field is populated with "results-ferran-aran" and has an "Info" link. A note below the field states: "Bucket name must be unique within the global namespace and follow the bucket naming rules. See rules for bucket naming". At the bottom of the configuration section, there is a section for "Copy settings from existing bucket - optional" with a "Choose bucket" button and the format "Format: s3://bucket/prefix".

## Writing files to S3 with python

We're now going to use the following code to write a file to the bucket we just created:

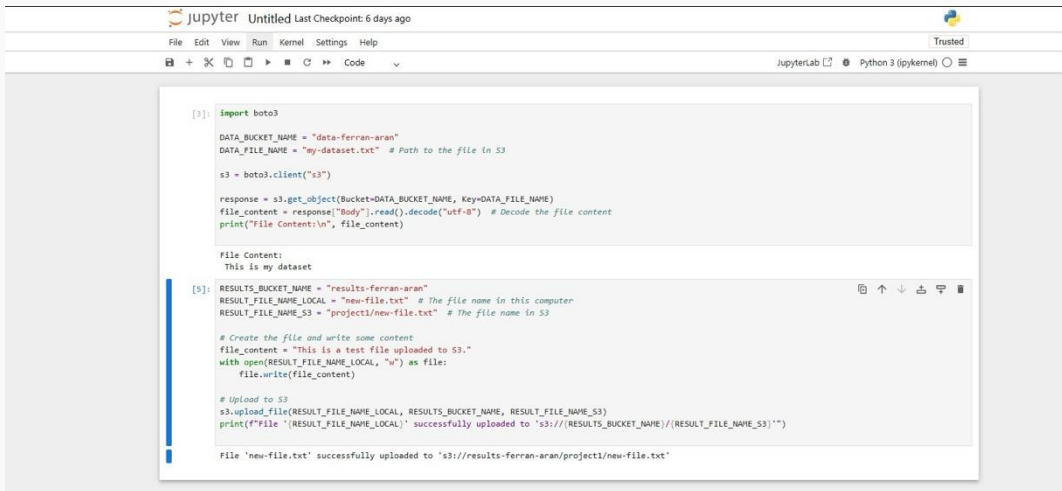
```
RESULTS_BUCKET_NAME = "results-your-name"
RESULT_FILE_NAME_LOCAL = "new-file.txt" # The file name in this computer
RESULT_FILE_NAME_S3 = "project1/new-file.txt" # The file name in S3

# Create the file and write some content
file_content = "This is a test file uploaded to S3."
with open(RESULT_FILE_NAME_LOCAL, "w") as file:
    file.write(file_content)

# Upload to S3
s3.upload_file(RESULT_FILE_NAME_LOCAL, RESULTS_BUCKET_NAME, RESULT_FILE_NAME_S3)
print(f"File '{RESULT_FILE_NAME_LOCAL}' successfully uploaded to 's3://{RESULTS_BUCKET_NAME}/{RESULT_FILE_NAME_S3}'")
```

# Writing files to S3 with python

If everything is correct you should see the following:



```
[3]: import boto3

DATA_BUCKET_NAME = "data-ferran-aran"
DATA_FILE_NAME = "my-dataset.txt" # Path to the file in S3

s3 = boto3.client("s3")

response = s3.get_object(Bucket=DATA_BUCKET_NAME, Key=DATA_FILE_NAME)
file_content = response["Body"].read().decode("utf-8") # Decode the file content
print("File Content:\n", file_content)

File Content:
This is my dataset

[5]: RESULTS_BUCKET_NAME = "results-ferran-aran"
RESULT_FILE_NAME_LOCAL = "new-file.txt" # The file name in this computer
RESULT_FILE_NAME_S3 = "project1/new-file.txt" # The file name in S3

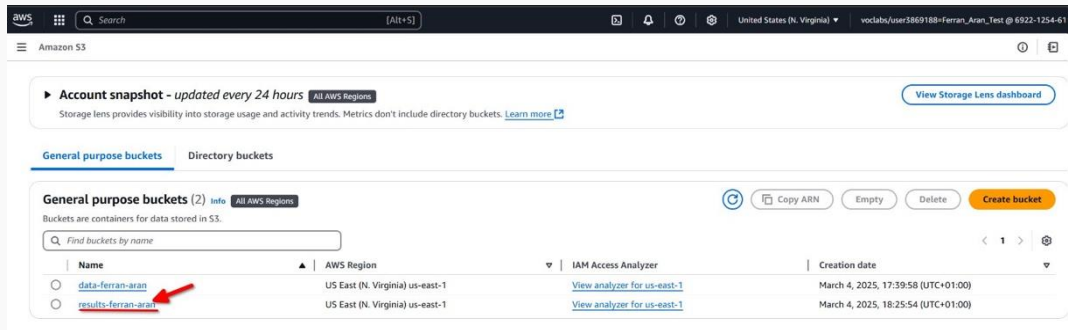
# Create the file and write some content
file_content = "This is a test file uploaded to S3."
with open(RESULT_FILE_NAME_LOCAL, "w") as file:
    file.write(file_content)

# Upload to S3
s3.upload_file(RESULT_FILE_NAME_LOCAL, RESULTS_BUCKET_NAME, RESULT_FILE_NAME_S3)
print(f"file '{RESULT_FILE_NAME_LOCAL}' successfully uploaded to 's3://{RESULTS_BUCKET_NAME}/{RESULT_FILE_NAME_S3}'")

File 'new-file.txt' successfully uploaded to 's3://results-ferran-aran/project1/new-file.txt'
```

# Inspecting the bucket from the AWS Console

We can now navigate to the S3 dashboard and click on the `results-{your-name}` bucket.



The screenshot shows the AWS S3 console interface. At the top, there's a navigation bar with the AWS logo, a search bar, and the text "[Alt+S]". Below that, the page title is "Amazon S3". A notification banner at the top left says "Account snapshot - updated every 24 hours" with a link to "View Storage Lens dashboard". The main content area is divided into "General purpose buckets" and "Directory buckets". Under "General purpose buckets", there's a sub-header "General purpose buckets (2)" and a search bar "Find buckets by name". Below the search bar is a table with columns: Name, AWS Region, IAM Access Analyzer, and Creation date. Two buckets are listed: "data-ferran-aran" and "results-ferran-aran". A red arrow points to the "results-ferran-aran" bucket. To the right of the table are buttons for "Copy ARN", "Empty", "Delete", and "Create bucket".

Name	AWS Region	IAM Access Analyzer	Creation date
<a href="#">data-ferran-aran</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	March 4, 2025, 17:39:58 (UTC+01:00)
<a href="#">results-ferran-aran</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	March 4, 2025, 18:25:54 (UTC+01:00)

# Inspecting the bucket from the AWS Console

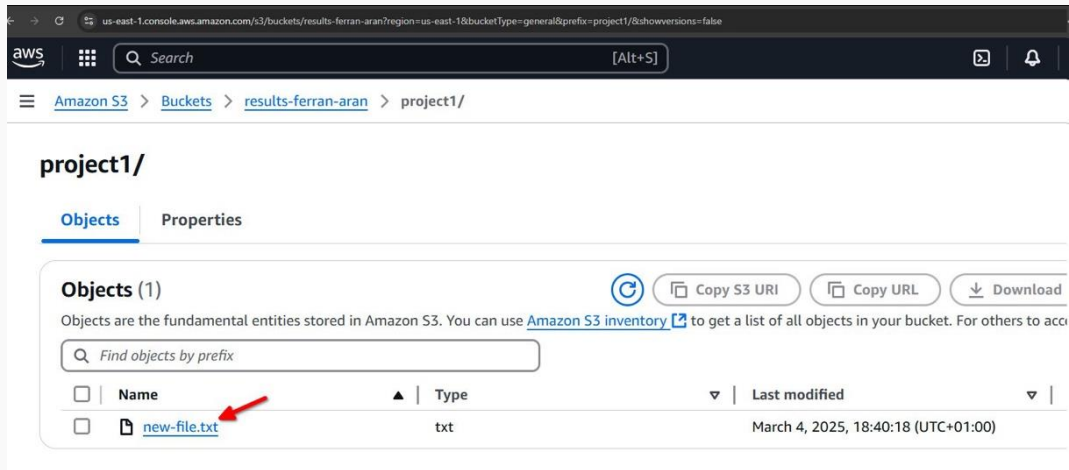
Inside the bucket we should see the folder we just created, click on it.

The screenshot shows the AWS S3 console interface. At the top, there's a search bar and navigation icons. Below that, the breadcrumb path is 'Amazon S3 > Buckets > results-ferran-aran'. The main heading is 'results-ferran-aran Info'. There are several tabs: 'Objects' (selected), 'Metadata', 'Properties', 'Permissions', 'Metrics', 'Management', and 'Access Points'. Under the 'Objects' tab, there's a section titled 'Objects (1)' with three buttons: 'Copy S3 URI', 'Copy URL', and 'Download'. Below this is a search input field with the placeholder text 'Find objects by prefix'. A table lists the objects with columns for 'Name', 'Type', 'Last modified', and 'Size'. The first row shows a folder named 'project1/' with a red arrow pointing to it.

<input type="checkbox"/>	Name	Type	Last modified	Size
<input type="checkbox"/>	<a href="#">project1/</a>	Folder	-	

# Inspecting the bucket from the AWS Console

And now we should see the file we just uploaded.



The screenshot shows the AWS S3 console interface. The breadcrumb navigation indicates the path: Amazon S3 > Buckets > results-ferran-aran > project1/. The main heading is 'project1/'. Below this, there are two tabs: 'Objects' (selected) and 'Properties'. The 'Objects (1)' section shows a list of objects. A search bar is present with the placeholder text 'Find objects by prefix'. The object list has columns for Name, Type, and Last modified. A red arrow points to the file 'new-file.txt' in the Name column. The file's type is 'txt' and it was last modified on 'March 4, 2025, 18:40:18 (UTC+01:00)'. At the top right of the object list, there are buttons for 'Copy S3 URI', 'Copy URL', and 'Download'.

us-east-1.console.aws.amazon.com/s3/buckets/results-ferran-aran?region=us-east-1&bucketType=general&prefix=project1/&showversions=false

aws Search [Alt+S]

Amazon S3 > Buckets > results-ferran-aran > project1/

## project1/

Objects Properties

### Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to acc

Find objects by prefix

<input type="checkbox"/>	Name	Type	Last modified
<input type="checkbox"/>	<a href="#">new-file.txt</a>	txt	March 4, 2025, 18:40:18 (UTC+01:00)

Copy S3 URI Copy URL Download

## **Sync with a bucket**

---

# Syncing a local directory to download from a bucket

## From the AWS CLI

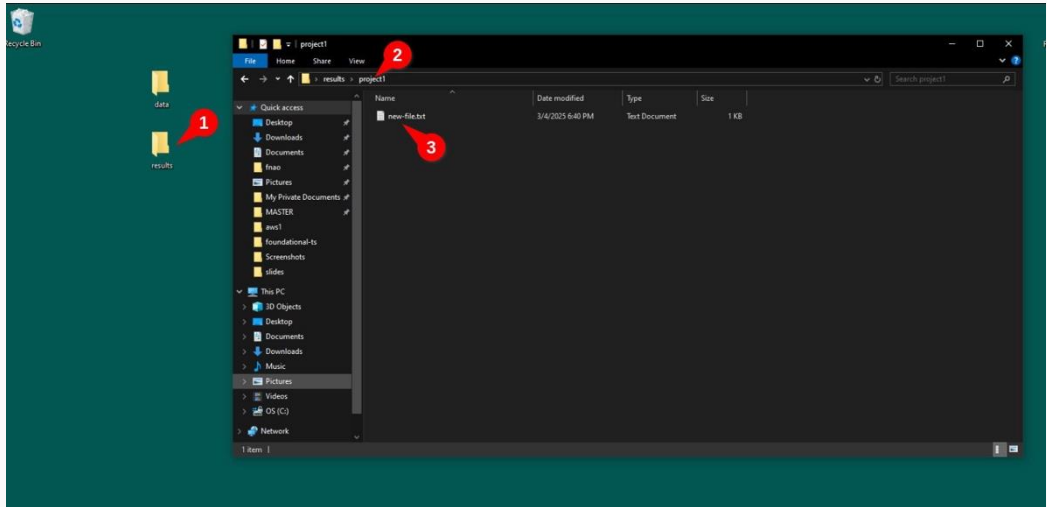
1. Create a new folder on your desktop and name it `results`.
2. Open a terminal and navigate to the folder. Use `cd Desktop`.
3. Run `aws s3 sync s3://results-{your-name} results` to download the files from the bucket.

In my case this is the result:

```
PS C:\Users\fnao\Desktop> aws s3 sync s3://results-ferran-aran results
download: s3://results-ferran-aran/project1/new-file.txt to results\project1\new-file.txt
PS C:\Users\fnao\Desktop>
```

## Syncing a local directory to download from a bucket

We can use the File Explorer to navigate to the folder and see the file we just downloaded.



## Recap

---

Today we have learned how to:

- Install the AWS CLI
- Configure AWS credentials
- Create an S3 bucket
- Sync a local directory to download from a bucket
- Sync a local directory to upload to a bucket
- Load files from the bucket to the notebook from python
- Write files from the notebook to the bucket from python

- With the setup we have done today we ended up with a bucket on which we can **upload datasets that we have to work on.**
- This datasets can be **accessed from any machine** with internet connection and the necessary permissions.
- We also have a results bucket which we can sync with our machine so **we have the results of our projects available locally.**

## Recap - Subject's website

- Remember there is a website with useful information related to the subject.
- It has recently been updated with information about past sessions.
- Two new guides have been added to *Get started with AWS* and *Set up your Lab for every session*.