

High Performance and Distributed Computing for Big Data

Unit 3: AWS - S3

Ferran Aran Domingo ferran.aran@udl.cat

Universitat Rovira i Virgili and Universitat de Lleida

- AWS S3: Storing files in the cloud

Today's lecture

- AWS S3: Storing files in the cloud
 - Installing AWS CLI

- AWS S3: Storing files in the cloud
 - Installing AWS CLI
 - Configuring AWS credentials

- AWS S3: Storing files in the cloud
 - Installing AWS CLI
 - Configuring AWS credentials
 - Creating an S3 bucket

- 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

- 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

- 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

- 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
 - Syncinc 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 `hdc-b-{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 on the AWS website. The page is titled "AWS CLI install and update instructions". It includes a navigation bar with links like "Get started", "Service guides", "Developer tools", and "AI resources". A search bar and a "Create an AWS Account" button are also visible. The main content area is divided into sections: "Important" (stating that AWS CLI versions 1 and 2 use the same 'aws' command name), "Topics" (listing links for install and update instructions, troubleshooting, and next steps), and "AWS CLI install and update instructions". Under the "Windows" section, there are "Install and update requirements" and "Install or update the AWS CLI" instructions. The "Install or update the AWS CLI" section includes a numbered list of steps, with the first step being to download and run the AWS CLI MSI installer. A red circle with the number "1" highlights the "Windows" section header, and a red arrow points to the command prompt code block.

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 `aws` 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.

1. 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 /s 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

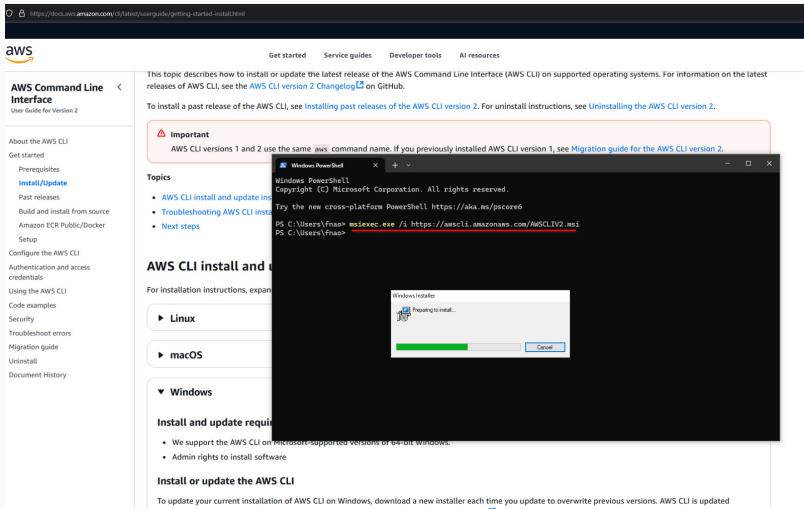
Did this page help you?

Yes No

Provide feedback

Installing the AWS CLI

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



The screenshot displays the AWS Command Line Interface (CLI) installation guide on the AWS website, overlaid with a Windows PowerShell terminal window showing the installation process.

AWS Command Line Interface
User Guide for Version 2

Important
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](#).

Topics

- [AWS CLI install and update instructions](#)
- [Troubleshooting AWS CLI installation](#)
- [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

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell <https://aka.ms/pscore6>
PS C:\Users\fnao> msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi
PS C:\Users\fnao>

Windows Installer
Preparing to install...

Installing the AWS CLI

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

The screenshot shows the AWS Command Line Interface User Guide for Version 2. The left sidebar contains a navigation menu with links like '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 is titled 'macOS' and includes a red callout '1' pointing to the section header. Below the header, there is a section 'Install and update requirements' with two bullet points. A table titled 'macOS version support matrix' lists supported versions. Below the table, there is a section 'Install or update the AWS CLI' with a red callout '2' pointing to the text. Under this section, there are tabs for 'GUI installer', 'Command line installer - All users', and 'Command line - Current user'. The 'Command line installer - All users' tab is selected, showing instructions and a terminal command block with a red arrow pointing to a copy icon. The terminal command is:

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

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.

Command line installer - All users

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.

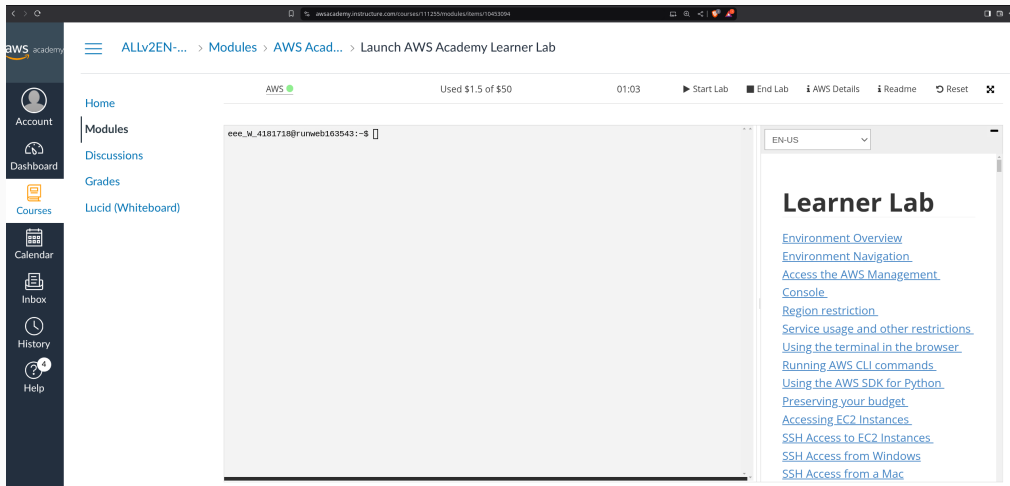
```
$ curl "https://awscli.amazonaws.com/AWSCLIV2.pkg" -o "AWSCLIV2.pkg"
$ sudo installer -pkg AWSCLIV2.pkg -target /
```

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 the AWS Academy Learner Lab interface. The browser address bar displays `awsacademy.instructure.com/courses/111255/modules/items/10433094`. The breadcrumb navigation path is `ALLv2EN-... > Modules > AWS Acad... > Launch AWS Academy Learner Lab`. The interface includes a left sidebar with navigation links: Home, Modules, Discussions, Grades, Lucid (Whiteboard), Account, Dashboard, Courses, Calendar, Inbox, History, and Help. The main content area features a terminal window with the prompt `eee_w_4181718@runweb163543:~$`. Above the terminal, it shows `AWS` with a green status indicator, `Used $1.5 of $50`, and a timer at `01:03`. Action buttons for `Start Lab`, `End Lab`, `AWS Details`, `Readme`, and `Reset` are present. On the right, a panel titled **Learner Lab** contains a list of links: [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](#), and [SSH Access from a Mac](#). A dropdown menu at the top of this panel is set to `EN-US`.

Configuring the AWS Credentials

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

The screenshot shows the AWS Academy Learner Lab interface. The top navigation bar includes a hamburger menu, the breadcrumb path "ALLv2EN-... > Modules > AWS Acad... > Launch AWS Academy Learner Lab", and a red callout "1" pointing to the "AWS Details" link in the top right. Below the navigation bar, the main content area is divided into a left sidebar and a main panel. The sidebar contains links for "Home", "Modules", "Discussions", "Grades", and "Lucid (Whiteboard)". The main panel has a header with "AWS" (with a green status dot), "Used \$1.6 of \$50", "01:01", and buttons for "Start Lab", "End Lab", "AWS Details", "Readme", "Reset", and a close icon. The "AWS Details" button is highlighted with a red callout "1". Below the header is a terminal window showing the prompt "eee_w_4181718@runweb163552:~\$". To the right of the terminal is a "Cloud Access" panel with a "Close" button. This panel contains the "AWS CLI:" section with a "Show" button, which is highlighted with a red callout "2". Below this is the "Cloud Labs" section showing session time and start/end dates. At the bottom of the panel, it shows the 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+ALE0a3
aws_session_token=IQoJb3JpZ2luX2VjELf////////wEaCXVzLXdlc3QtMiJGMEQCIE00uLiVzFVLHv5EbZPZj8hPqPJGou84lGfM0ezfiSP0AiB+S2vrdPaz4TFMEYXwe1e0Hr/C05m0B3N72MZHQ1n0DyqZAgjw////////8BEAEaDDY5MjIxMjU0NjExMiIMCDFeGSKhijJKhgHCKocCd6djgX83VD2PG83XsoMK972gcLP6P8T0ZvujiY4RToS2uqbjAW/2cNwdpukyH8LX0zAjI00CzpWr+M5HrC/7vUPZSAN0nVQdJvmbBrXZ+ln8SsjTuBFq1zD6bRhnV7iHhA8naNoXE4eNAM0C09HQ6JmFVAoPTQ3nGDPGKEfIPmG6KzIR3H0vmuZSvsRPb4MctQxk/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.
```

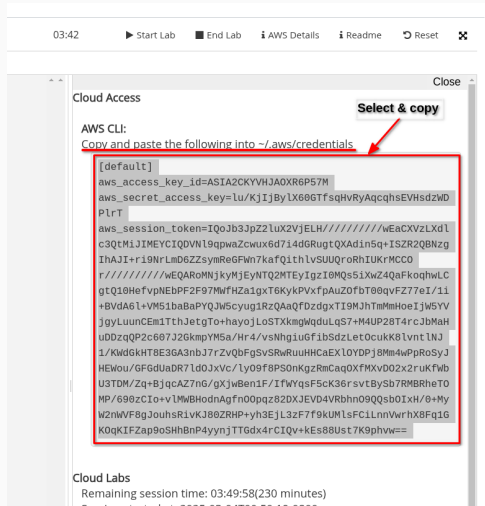
or for MacOS users:

```
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.



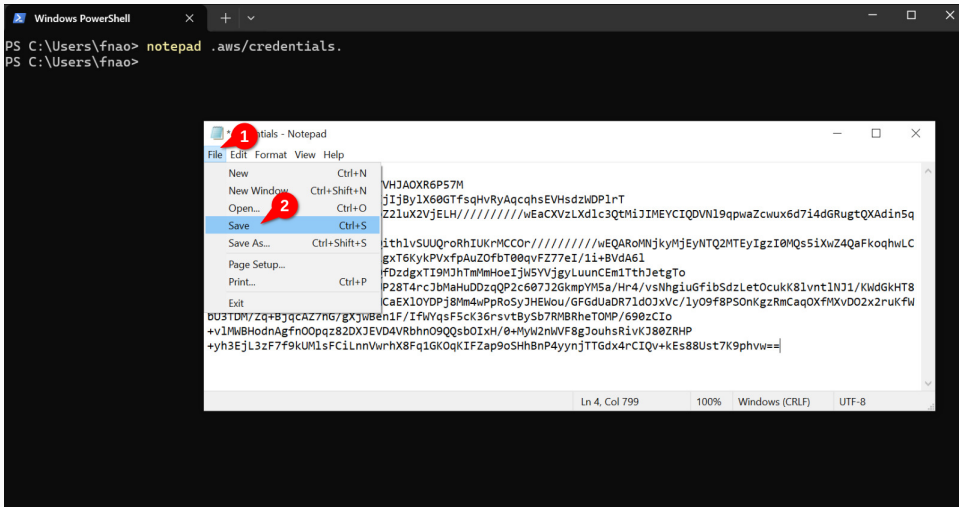
The screenshot shows the 'Cloud Access' panel in the AWS Academy interface. At the top, there is a navigation bar with '03:42', 'Start Lab', 'End Lab', 'AWS Details', 'Readme', 'Reset', and a close icon. The 'Cloud Access' panel has a 'Close' button in the top right corner. Below the title, it says 'AWS CLI:' and 'Copy and paste the following into ~/.aws/credentials'. A red box highlights the credentials text, and a red arrow points to it with the label 'Select & copy'.

```
[default]
aws_access_key_id=ASIA2CKYVHJA0XR6P57M
aws_secret_access_key=lu/KjIjByLx60GTfsqHvRyAqcqhsEVHsdzWD
P1rT
aws_session_token=IQoJb3JpZ2luX2VjELH////////wEaCXVzLXdL
c3QtMiJIMEYCIQDVNl9qpwaZcwux6d7i4dGRugtQXAdin5q+ISZR2QBnZg
IhAJI+r19NrLmD6ZZsymReGFwn7kafQ1thlvSUUQroRHIUKrMCCO
r////////wEQARoMNjkyMjEyNTQ2MTEyIgzI0MQs5iXwZ4QaFkoqhWLC
gtQ10HefvpNEbPF2F97MwfHza1gxT6KyKpVxfpAuZ0fbT00qvFZ77eI/1i
+BVdA6l+VM51baBaPYQJw5cyug1RzQAaQfDzdgxTI9MJhTmMmHoeIjw5YV
jgyLuunCEm1TthJtgTo+hayojLoSTXkmgWqduLqS7+M4UP28T4rcJbMaH
uDDzqQP2c607J2GkmpYM5a/Hr4/vsNhgiuGfibSdzLet0cukK8lvntlNJ
1/KwdGkHT8E3GA3nbJ7rZvQbFgSvSRwRuHHCaEXlOYDPj8Mm4wPpRoSyJ
HEWou/GFGdUaDR7ld0JxVc/ly09f8P50nKgZrmCaqOXfMXvD02x2ruKfwb
U3TDM/Zq+BjqcAZ7nG/gXjwBen1F/IffWYqsF5cK36rsvtBySb7RMBRheTO
MP/690zcIo+vLMwBhodnAgfn00pqz82DXJEVD4VRbhn09Qqsb0IxH/0+My
w2nlwVF8gJouhsR1vKJ00ZRHPhy3EjL3zF7f9kUmlsFC1LnnVvrhX8Fq1G
K0qKIFZap9oSHhBnP4yynjTTGdx4rCIQv+kEs88Ust7K9phvw==
```

Cloud Labs
Remaining session time: 03:49:58(230 minutes)
Session started at 2025-03-04T00:50:10.0000

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.



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=ASIA2CKYVHJA0XR6P57M
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": "AROA2CKYVHJALK46ZMHVM:user3869188=Ferran_Aran_Test",
  "Account": "692212546112",
  "Arn": "arn:aws:sts::692212546112:assumed-role/voclabs/user3869188=Ferran_Aran_Test"
}

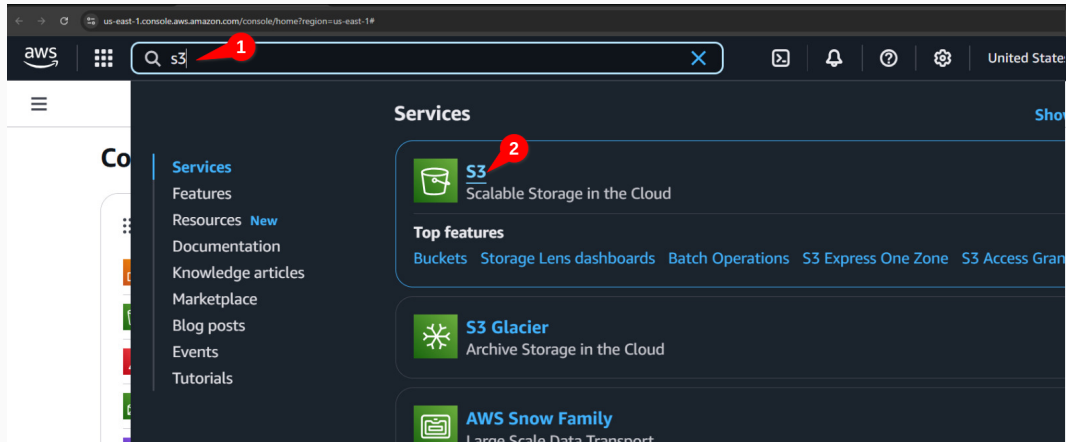
PS C:\Users\fnao>
```

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's a navigation bar with the AWS logo, a search bar, and various icons. Below this, the left sidebar contains the 'Amazon S3' menu with options like 'General purpose buckets', 'Directory buckets', 'Table buckets', 'Access Grants', 'Access Points', 'Object Lambda Access Points', 'Multi-Region Access Points', 'Batch Operations', and 'IAM Access Analyzer for S3'. The main content area is titled 'General purpose buckets (1)' and includes a search bar with the placeholder 'Find buckets by name'. Below the search bar, there's a table with columns for 'Name', 'AWS Region', and 'IAM Access Analyzer'. The table lists one bucket named 'ferran-test123' in the 'US East (N. Virginia) us-east-1' region. To the right of the table, there are buttons for 'Copy ARN', 'Empty', 'Delete', and 'Create bucket'. The 'Create bucket' button is highlighted with a red circle and the number 1.

aws [Search] [Alt+S] United States (N. Virginia) voclabs/user3869188=Ferran_Aran_Test @ 6922-1254-6112

Amazon S3

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

▼ Storage Lens

► **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 (1) Info All AWS Regions

Buckets are containers for data stored in S3.

Find buckets by name

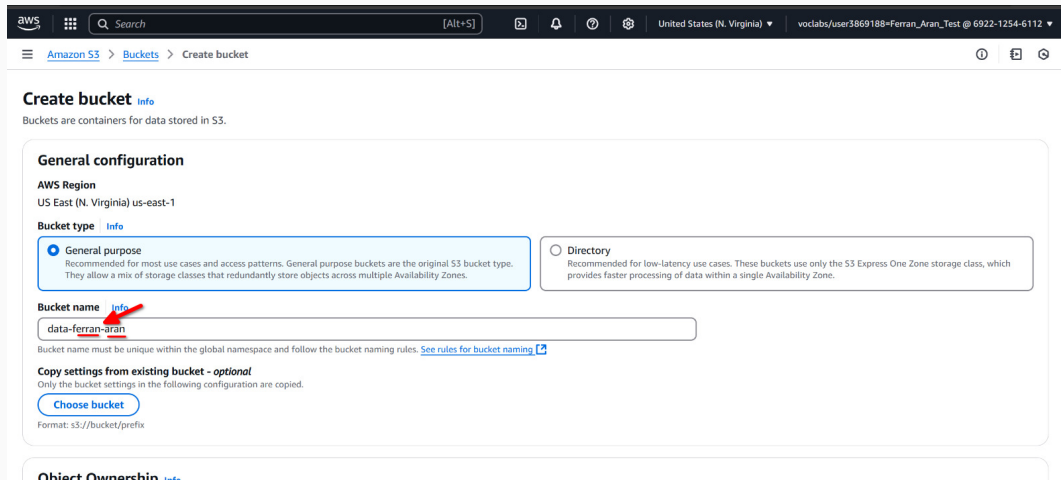
< 1 > ⚙

Name	AWS Region	IAM Access Analyzer
☐ ferran-test123	US East (N. Virginia) us-east-1	View analyzer for us-east-1

Copy ARN Empty Delete **Create bucket**

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](#)

data-ferran-aran

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.

[Choose bucket](#)

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 ⓘ

Server-side encryption is automatically applied to new objects stored in this bucket.

Encryption type ⓘ

☒ 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

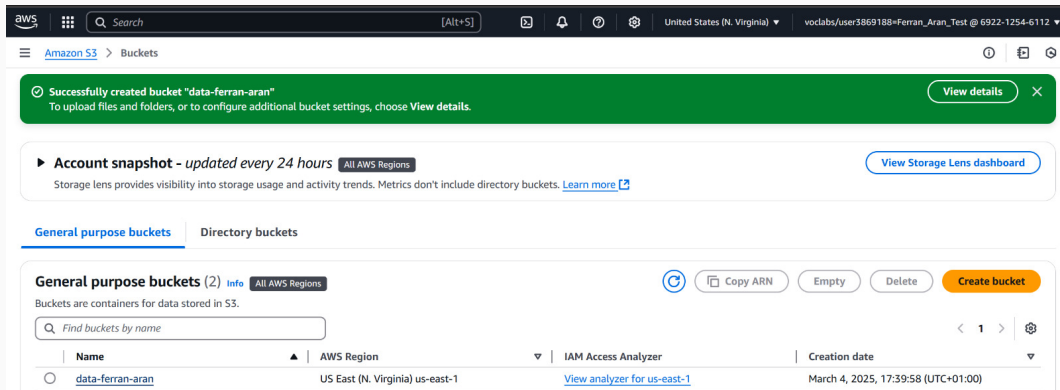
► Advanced settings

ⓘ 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 various utility icons. Below the navigation bar, the breadcrumb trail shows 'Amazon S3 > Buckets'. A prominent green success message banner at the top states: 'Successfully created bucket "data-ferran-aran"'. Below this, there's a section for 'Account snapshot' with a 'View Storage Lens dashboard' button. 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)' with an 'info' icon and a region filter 'All AWS Regions'. A description states 'Buckets are containers for data stored in S3.' Below this is a search bar 'Find buckets by name'. A table lists the buckets with columns for Name, AWS Region, IAM Access Analyzer, and Creation date. One bucket is listed: 'data-ferran-aran' in 'US East (N. Virginia) us-east-1', with a link to 'View analyzer for us-east-1' and a creation date of 'March 4, 2025, 17:39:58 (UTC+01:00)'. Action buttons like 'Copy ARN', 'Empty', 'Delete', and 'Create bucket' are visible at the top of the bucket list.

aws Search [Alt+S] United States (N. Virginia) voclabs/user3869188=Ferran_Aran_Test @ 6922-1254-6112

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
data-ferran-aran	US East (N. Virginia) us-east-1	View analyzer for us-east-1	March 4, 2025, 17:39:58 (UTC+01:00)

Uploading a file to the bucket

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.

Sync with a bucket

Syncing a local directory to upload to a bucket

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 folder. 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 navigation bar with the AWS logo, a search bar, and utility icons. Below this, the 'Amazon S3' header is visible. On the left, a sidebar lists various S3 features: General purpose buckets, Directory buckets, Table buckets, Access Grants, Access Points, Object Lambda Access Points, Multi-Region Access Points, Batch Operations, and IAM Access Analyzer for S3. The main content area is divided into two tabs: 'General purpose buckets' (active) and 'Directory buckets'. Under the 'General purpose buckets' tab, there's a section titled 'General purpose buckets (2)' with an 'Info' link and a filter for 'All AWS Regions'. Below this, a search bar prompts 'Find buckets by name'. A table lists the buckets with columns for 'Name', 'AWS Region', and 'IAM Access Analyzer'. The first bucket, 'data-ferran-aran', is highlighted with a red arrow. The table also shows the region as 'US East (N. Virginia) us-east-1' and a link to 'View analyzer for us-east-1'.

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

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

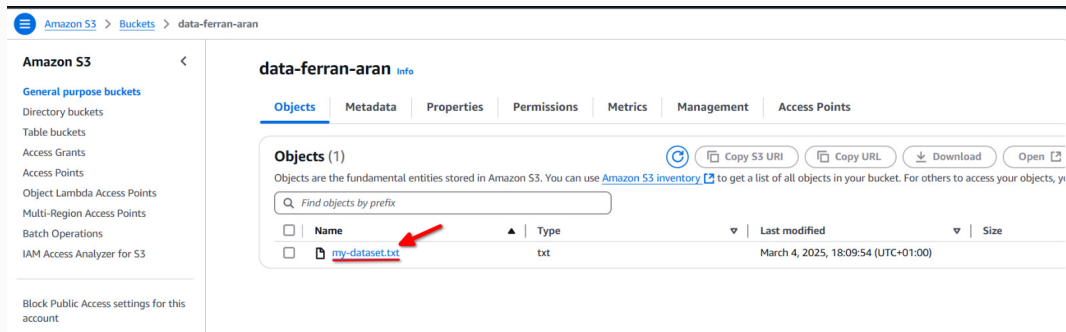
General purpose buckets (2) Info All AWS Regions
Buckets are containers for data stored in S3.

Find buckets by name

	Name	AWS Region	IAM Access Analyzer
☐	data-ferran-aran	US East (N. Virginia) us-east-1	View analyzer for us-east-1

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 displays the AWS S3 console interface for the bucket 'data-ferran-aran'. The left sidebar shows the 'Amazon S3' menu with options like 'General purpose buckets', 'Directory buckets', 'Table buckets', 'Access Grants', 'Access Points', 'Object Lambda Access Points', 'Multi-Region Access Points', 'Batch Operations', and 'IAM Access Analyzer for S3'. The main content area shows the bucket's details, including tabs for 'Objects', 'Metadata', 'Properties', 'Permissions', 'Metrics', 'Management', and 'Access Points'. The 'Objects' tab is active, displaying a list of objects. A search bar is present above the list. The list contains one object, 'my-dataset.txt', which is highlighted with a red arrow. The object's type is 'txt' and its last modified date is 'March 4, 2025, 18:09:54 (UTC+01:00)'. Above the list, there are buttons for 'Copy S3 URI', 'Copy URL', 'Download', and 'Open'.

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)   Copy S3 URI  Copy URL  Download  Open

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 [Object Lambda](#).

Find objects by prefix

☐	Name	Type	Last modified	Size
☐	 <u>my-dataset.txt</u>	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 left sidebar contains the 'Amazon S3' menu with options like 'General purpose buckets', 'Directory buckets', 'Table buckets', 'Access Grants', 'Access Points', 'Object Lambda Access Points', 'Multi-Region Access Points', 'Batch Operations', and 'IAM Access Analyzer for S3'. Below this is the 'Storage Lens' section with 'Dashboards', 'Storage Lens groups', and 'AWS Organizations settings'. The main content area is titled 'my-dataset.txt' and has tabs for 'Properties', 'Permissions', and 'Versions'. The 'Properties' tab is active, showing an 'Object overview' section with the following details: Owner (awslabsc0w4907751t1669516777), 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), and Key (my-dataset.txt). To the right of the overview is a section with three items: 'S3 URI' (s3://data-ferran-aran/my-dataset.txt, highlighted with a red box and a red arrow), 'Amazon Resource Name (ARN)' (arn:aws:s3:::data-ferran-aran/my-dataset.txt), and 'Entity tag (Etag)' (0fee9acadfe8753f721d44b985a90c67). At the bottom of this section is the 'Object URL' (https://data-ferran-aran.s3.us-east-1.amazonaws.com/my-dataset.txt). At the top right of the main content area are buttons for 'Copy S3 URI', 'Download', 'Open', and 'Object actions'.

Amazon S3 > Buckets > data-ferran-aran > my-dataset.txt

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

Storage Lens

Dashboards

Storage Lens groups

AWS Organizations settings

my-dataset.txt

Properties Permissions Versions

Object overview

Owner
awslabsc0w4907751t1669516777

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

S3 URI
<s3://data-ferran-aran/my-dataset.txt>

Amazon Resource Name (ARN)
<arn:aws:s3:::data-ferran-aran/my-dataset.txt>

Entity tag (Etag)
[0fee9acadfe8753f721d44b985a90c67](#)

Object URL
<https://data-ferran-aran.s3.us-east-1.amazonaws.com/my-dataset.txt>

Copy S3 URI Download Open Object actions

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*.

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*.

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.

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*.

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.

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`.

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`.

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



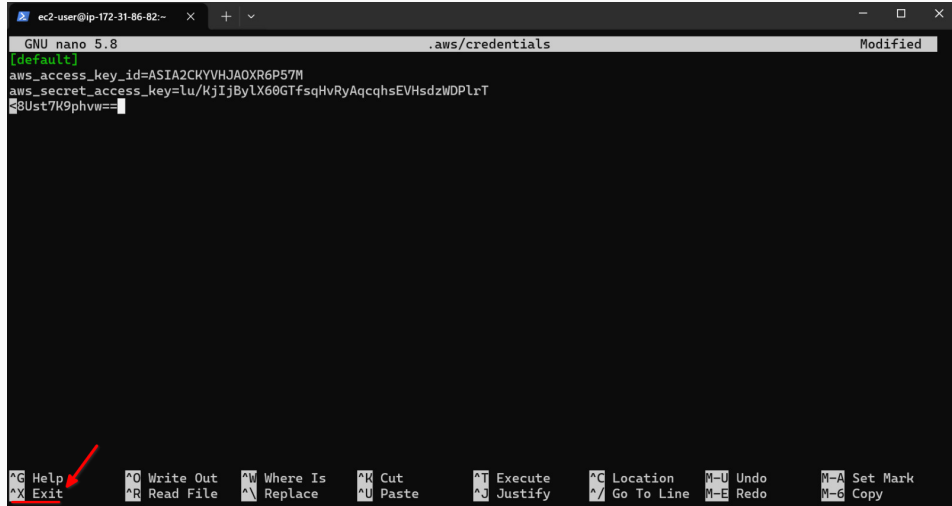
The image shows a terminal window with a dark background. The title bar at the top indicates the user is 'ec2-user' on an IP address of '172-31-86-92'. The terminal itself shows the 'GNU nano 5.8' editor interface. The current file being edited is '.aws/credentials', and the status bar on the right says 'Modified'. The main area of the terminal is empty, suggesting the file is new or has been cleared. At the bottom, there is a comprehensive list of nano editor shortcuts, including Help, Write Out, Where Is, Cut, Execute, Location, Undo, Set Mark, Exit, Read File, Replace, Paste, Justify, Go To Line, Redo, and Copy.

```
ec2-user@ip-172-31-86-92:~$ nano .aws/credentials
```

GNU nano 5.8 .aws/credentials Modified

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

Configuring AWS credentials on an EC2 instance



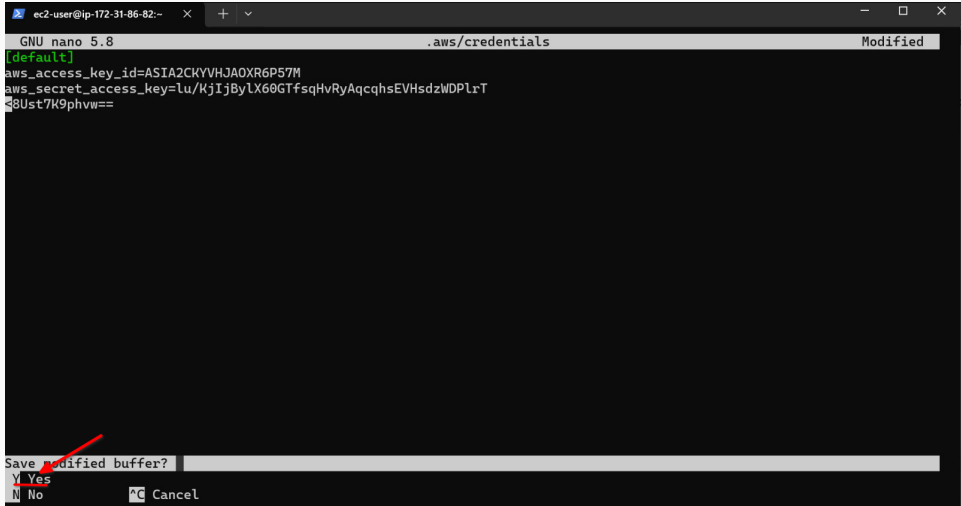
The screenshot shows a terminal window with a nano editor editing a file named `.aws/credentials`. The file content is as follows:

```
GNU nano 5.8 .aws/credentials Modified
[default]
aws_access_key_id=ASIA2CKYVHJA0XR6P57M
aws_secret_access_key=lu/KjIjBylX60GTfsqHvRyAqcqhsEVHsdzWDPlrT
8Ust7K9phvw==
```

The terminal window title bar indicates the user is `ec2-user` on an instance with IP `ip-172-31-86-82`. The nano editor's status bar at the bottom displays various keyboard shortcuts. A red arrow points to the `^X Exit` option on the left side of the status bar.

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

Configuring AWS credentials on an EC2 instance

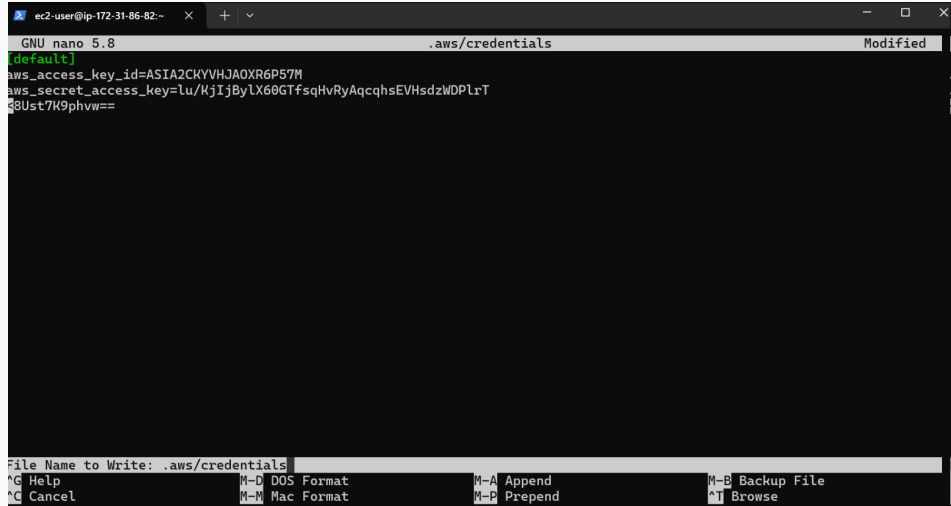


The screenshot shows a terminal window titled "ec2-user@ip-172-31-86-82:~". The terminal is running the GNU nano 5.8 editor, editing the file ".aws/credentials". The file content is as follows:

```
[default]
aws_access_key_id=ASIA2CKYVHJA0XR6P57M
aws_secret_access_key=lu/KjIjBylX60GTfsqHvRyAqcqhsEVHsdzWDPlrT
8Ust7K9phvw==
```

At the bottom of the terminal, a prompt "Save modified buffer?" is displayed. The "Y" option (Yes) is selected, indicated by a red arrow. The "N" option (No) and the "Ctrl+C" Cancel option are also visible.

Configuring AWS credentials on an EC2 instance



The screenshot shows a terminal window titled "ec2-user@ip-172-31-86-82:~". Inside the terminal, the GNU nano 5.8 text editor is open, editing the file ".aws/credentials". The file content is as follows:

```
[default]
aws_access_key_id=ASIA2CKYVHJA0XR6P57M
aws_secret_access_key=lu/KjIjBylX60GTfsqHvRyAqcqhsEVHsdzWDPLrT
s8Ust7K9phvw==
```

The bottom of the terminal shows the nano editor's status bar with the following information:

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

Configuring AWS credentials on an EC2 instance

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

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": "AROA2CKYVHJALK46ZMHVM: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 be using 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

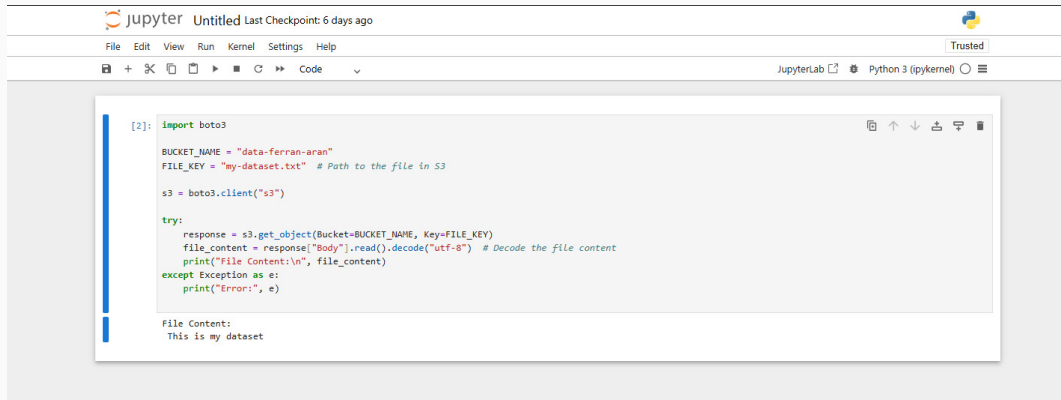
s3 = boto3.client("s3")

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. At the top, the title bar says "jupyter Untitled Last Checkpoint: 6 days ago". Below it is a menu bar with "File", "Edit", "View", "Run", "Kernel", "Settings", and "Help". To the right of the menu bar is a "Trusted" button. Below the menu bar is a toolbar with icons for file operations and a "Code" button. The main area contains a code cell with the following Python code:

```
[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)
```

Below the code cell, the output is displayed:

```
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).

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.

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.

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 that 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`.

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](#)

results-ferran-aran

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.

[Choose bucket](#)

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:

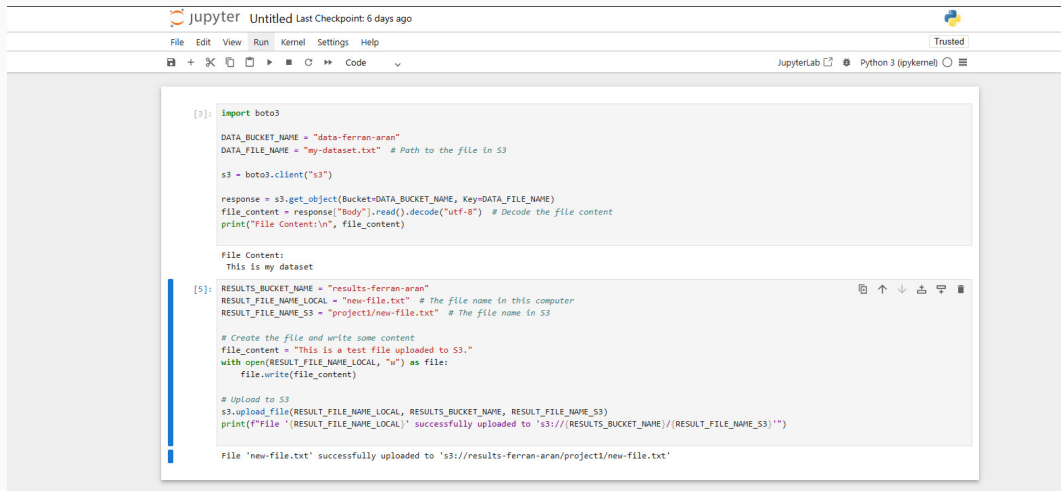
```
RESULTS_BUCKET_NAME = "results-your-name"
RESULT_FILE_NAME_LOCAL = "new-file.txt" # The file name in this computer
RESULT_FILE_NAME_S3 = "project1/new-file.txt" # The file name in S3

# 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

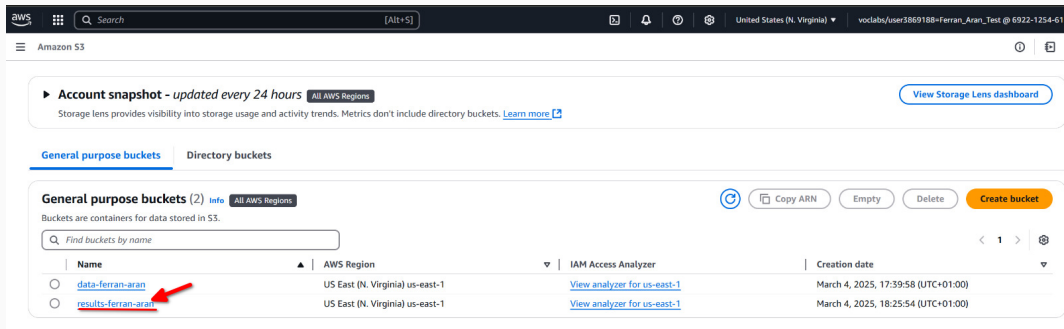
# Create the file and write some content
file_content = "This is a test file uploaded to S3."
with open(RESULT_FILE_NAME_LOCAL, "w") as file:
    file.write(file_content)

# 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 various utility icons. Below this, the 'Amazon S3' section is visible. A banner for 'Account snapshot' is present, followed by a 'View Storage Lens dashboard' button. 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)' with an 'Info' link and a 'All AWS Regions' button. Below this, a search bar prompts 'Find buckets by name'. A table lists the buckets:

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

Buttons for 'Copy ARN', 'Empty', 'Delete', and 'Create bucket' are located above the table. A red arrow points to the 'results-ferran-aran' bucket name.

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 navigation bar with the AWS logo, a search bar, and a [Alt+S] shortcut. Below the navigation bar, the breadcrumb trail reads: Amazon S3 > Buckets > results-ferran-aran. The main heading is 'results-ferran-aran' with an 'Info' link. Below the heading 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 these buttons is a text description: '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 yo'. Below the text is a search bar with the placeholder text 'Find objects by prefix'. Below the search bar is a table with the following columns: Name, Type, Last modified, and Size. The table contains one row with the name 'project1/' (highlighted by a red arrow), Type 'Folder', Last modified '-', and Size '-'. The 'Name' column has a checkbox and a folder icon next to the name.

aws Search [Alt+S]

Amazon S3 > Buckets > results-ferran-aran

results-ferran-aran [Info](#)

[Objects](#) Metadata Properties Permissions Metrics Management Access Points

Objects (1)

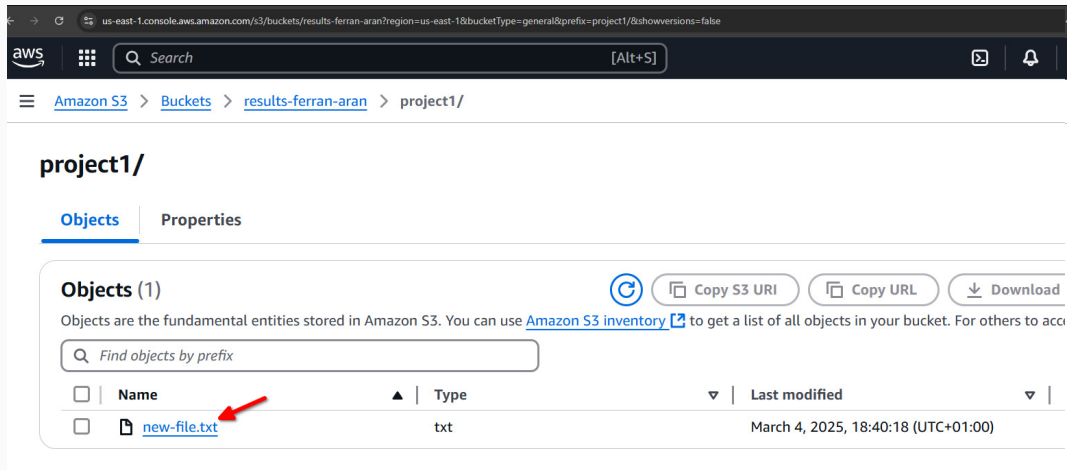
[Refresh](#) [Copy S3 URI](#) [Copy URL](#) [Download](#)

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 yo

☐	Name	Type	Last modified	Size
☐	project1/	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 at the top 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 red arrow points to the file [new-file.txt](#) in the **Name** column. The **Type** column shows 'txt' and the **Last modified** column shows 'March 4, 2025, 18:40:18 (UTC+01:00)'. Above the list, there are buttons for [Copy S3 URI](#), [Copy URL](#), and [Download](#). A search bar with the placeholder 'Find objects by prefix' is also present.

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

☐	Name	Type	Last modified
☐	new-file.txt	txt	March 4, 2025, 18:40:18 (UTC+01:00)

Sync with a bucket

Syncing a local directory to download from a bucket

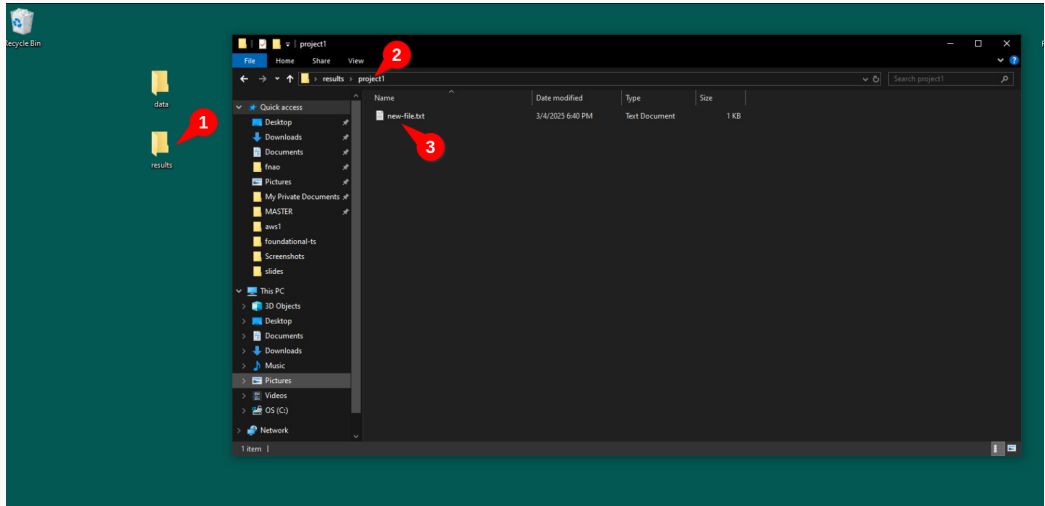
1. Create a new folder on your desktop and name it `results`.
2. Open a terminal and navigate to the folder. Use `cd Desktop`.
3. Run `aws s3 sync s3://results-{your-name} results` to download the files from the bucket.

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*.