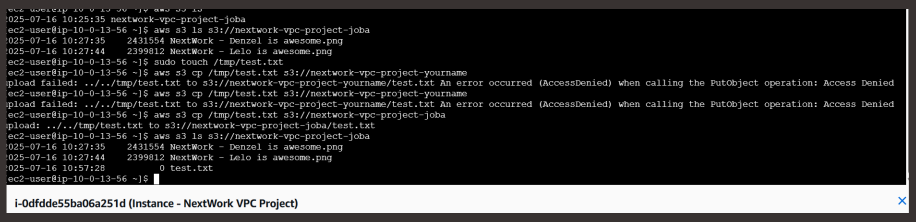
nextwork.org

Access S3 from a VPC

Joba Amunigun
https://www.linkedin.com/in/dvoice




Joba Amunigun
NextWork Student

Introducing Today's Project!

What is Amazon VPC?

Amazon VPC (Virtual Private Cloud) is a service that lets you create your own private network inside the AWS cloud — just like designing your own mini internet. It's useful because it gives you full control over how your resources connect and communicate, including things like: Public and private subnets Security settings Internet access Traffic routing With VPC, I can safely launch and manage AWS services (like EC2 instances or databases) in an isolated and secure environment, just the way I want them.

How I used Amazon VPC in this project

In today's project, I used Amazon VPC to build a secure network for my EC2 instance and S3 bucket to communicate. I configured the VPC, created a public subnet, attached an internet gateway, and set up the right route tables and security settings. Then, I tested everything by connecting my EC2 instance to my S3 bucket using the AWS CLI — uploading and listing files to confirm everything was working. This hands-on setup helped me understand how networking, permissions, and service integration work together in the cloud.

One thing I didn't expect in this project was...

One thing I didn't expect was how sensitive and specific access keys are. Even though I had configured my EC2 instance with access keys correctly, I still couldn't upload files to S3 — because the keys didn't have the right permissions attached. It taught me that using access keys isn't just about having them — it's about making sure they're paired with the correct IAM policies to do the job.

This project took me...

This project took me a little over 3 hours, including hands-on setup and deep study

In the first part of my project...

Step 1 - Architecture set up

In this step, I'm going to create a custom VPC from scratch to give me full control over my networking environment. Then, I'll launch an EC2 instance into that VPC so I can run applications securely within my own isolated network. This setup is important because it mimics how real companies build secure cloud infrastructure from the ground up.

Step 2 - Connect to my EC2 instance

In this step, I'm going to connect directly to my EC2 instance so I can start using it just like I would a remote computer. The goal is to access the command line and use the instance to interact with other AWS services — this helps simulate how cloud resources work together in a real-world environment.

Step 3 - Set up access keys

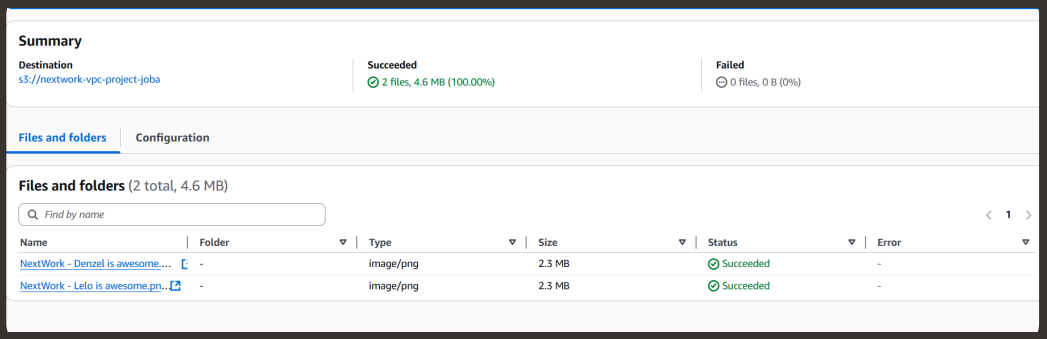
In this step, I'm creating access keys so my EC2 instance can securely connect to and interact with AWS services — like S3 buckets or databases — without me logging in manually. These access keys act as credentials that let the instance perform tasks programmatically, which is essential for automating workflows and allowing applications to talk to other AWS resources behind the scenes.

Joba Amunigun
NextWork Student

Architecture set up

I started my project by creating my own Virtual Private Cloud (VPC) with 1 Availability Zone to keep things simple and focused. I set up 1 public subnet within that AZ to host resources that need internet access. After setting up the VPC and subnet, I launched an EC2 instance into the public subnet — this acts as a virtual server that I can use to interact with other AWS services. For now, I didn't create any private subnets, since the goal was to build a straightforward, functional network foundation.

I also set up an Amazon S3 bucket to store files in the cloud. After creating the bucket, I uploaded two files from my local computer to it — which means the bucket is now holding actual objects. This sets the stage for connecting my EC2 instance to S3 and practicing how to access, list, or manage cloud storage from a server. It's a key step in learning how cloud services talk to each other



Joba Amunigun
NextWork Student

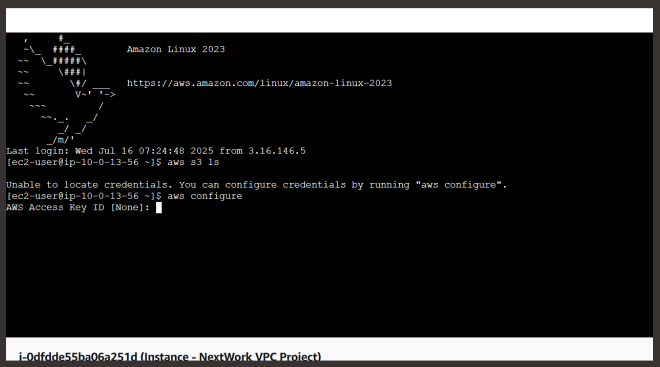
Running CLI commands

AWS CLI is the Amazon Web Services Command Line Interface — a tool that lets you manage and interact with AWS services directly from your terminal, instead of clicking through the web console. You can create, configure, and control resources like EC2, S3, IAM, and VPCs — all through typed commands. I have access to AWS CLI because AWS provides a tool called CloudShell — a built-in terminal in the AWS Console that comes with the CLI pre-installed. This makes it easy to practice AWS CLI commands without needing any local setup. It's powerful for automation, speed, and understanding what's happening behind the scenes in your cloud environment.

The first command I ran was " aws s3 ls " This command is used to list the S3 buckets in your account (ls stands for list!) - Wasn't a success because i needed to provide credentials (Credentials in AWS are like keys that let you access and manage AWS services securely)

The second command I ran was "aws configure". This command is used to set up the AWS CLI with your access credentials, default region, and output format. It's like logging in — it tells the CLI who you are and how to connect to your AWS account. Once configured, every command I run in the CLI uses these credentials to interact with AWS services securely.

Joba Amunigun
NextWork Student




Joba Amunigun
NextWork Student

Access keys

Credentials

To set up my EC2 instance to interact with my AWS environment, I configured it using the aws configure command. This command let me securely enter my Access Key ID, Secret Access Key, default AWS region, and output format — so that my instance could start running AWS CLI commands like accessing S3. It's like logging in behind the scenes, so my EC2 instance knows who I am and what resources I'm allowed to work with.

Access keys are a pair of credentials (an Access Key ID and a Secret Access Key) that allow you or your applications to interact with AWS programmatically — through the AWS CLI, SDKs, or APIs. They act like a username and password for your AWS account, but specifically for tools and scripts — not for logging into the AWS web console. Think of access keys as the keys you give to software or servers so they can securely talk to your AWS services.

Secret access keys are the "password" that pairs with your access key ID (your AWS "username") — together, they form the full credentials needed to access AWS services programmatically through the CLI or APIs. Secret is the key word — because anyone who has your secret access key can potentially access your AWS account, so it's important to store it securely and never share or expose it.

Best practice

Although I'm using access keys in this project, a best practice alternative is to use IAM roles — especially when working with EC2 instances. IAM roles allow EC2 instances to inherit permissions securely, without needing to store access keys. You can easily attach or detach roles from instances, and AWS handles the credential rotation behind the scenes — which makes it safer and easier to manage. We're using access keys here to understand how they work, but in real-world cloud environments, IAM roles are the more secure and scalable option

In the second part of my project...

Step 4 - Set up an S3 bucket

In this step, I'm creating a bucket in Amazon S3, which is a cloud storage service used to store files, folders, and other data. We're setting up the bucket now so that my EC2 instance can later access it — for example, to list, upload, or download objects from the cloud. This step is important because it connects storage and compute services together — a common real-world task in cloud engineering!

Step 5 - Connecting to my S3 bucket

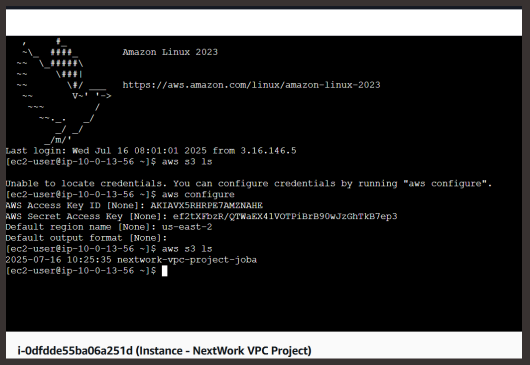
In this step, I'm going back into my EC2 instance to test if it can connect to and interact with my S3 bucket — like listing the files I uploaded earlier. This is important because it shows how different AWS services can work together securely — a key skill in cloud engineering. I'll be using the AWS CLI inside EC2 to talk to my S3 bucket, just like a real application would.

Joba Amunigun
NextWork Student

Connecting to my S3 bucket

The first command I ran was " aws s3 ls " This command is used to list the S3 buckets in your account (ls stands for list!) - Wasn't a success because i needed to provide credentials (Credentials in AWS are like keys that let you access and manage AWS services securely)

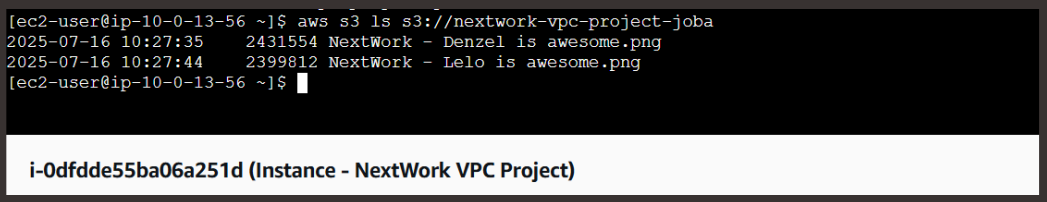
When I ran the command "aws s3 ls" again, the terminal responded with:"2025-07-16 10:25:35 nextwork-vpc-project-joba". This indicated that my EC2 instance successfully connected to my S3 bucket and was able to list its contents — confirming that my credentials and configuration were set up correctly.



Joba Amunigun
NextWork Student

Connecting to my S3 bucket

Another CLI command I ran was:"aws s3 ls s3://nextwork-vpc-project-joba" which returned: 2025-07-16 10:27:35 2431554 NextWork - Denzel is awesome.png 2025-07-16 10:27:44 2399812 NextWork - Lelo is awesome.png This confirmed that my EC2 instance was able to successfully access the contents of my S3 bucket — and list the exact files I uploaded earlier from the console. Super satisfying!



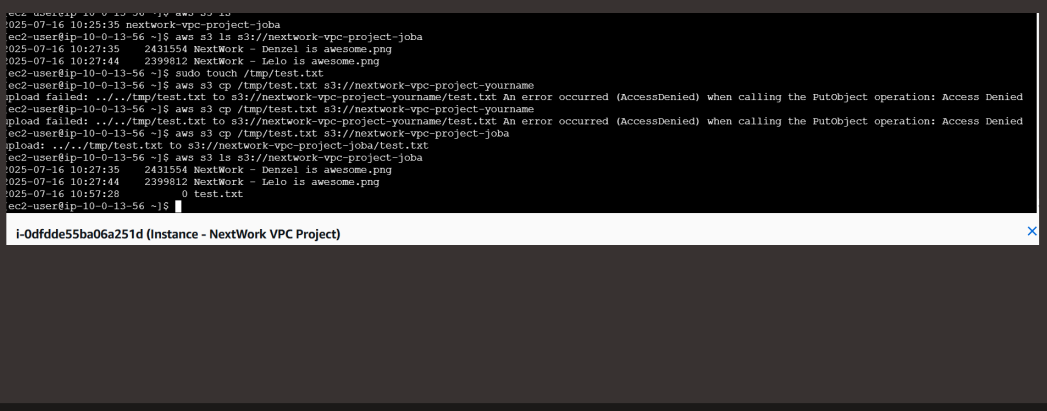
Joba Amunigun
NextWork Student

Uploading objects to S3

To upload a new file to my bucket, I first ran the command:"sudo touch /tmp/test.txt" This command creates a blank file named test.txt inside the /tmp/ directory of my EC2 instance. It's a simple way to generate a file that I can later upload to S3 for testing permissions and connectivity.

The second command I ran was:"aws s3 cp /tmp/test.txt s3://nextwork-vpc-project-joba" This command will upload the test.txt file from my EC2 instance to my S3 bucket. It’s how I tested whether my instance had the correct permissions to write to the bucket using the AWS CLI.

The third command I ran was:"aws s3 ls s3://nextwork-vpc-project-joba" which validated that the file upload was successful — I could now see test.txt listed in the bucket. This confirmed that my EC2 instance had proper connectivity and permissions to interact with Amazon S3.



nextwork.org

The place to learn & showcase your skills

Check out nextwork.org for more projects