<https://aws.plainenglish.io/mastering-cost-optimisation-with-shell-scripting-automate-log-storage-in-s3-for-budget-friendly-689ef5c6ba5a>



## Intro

Welcome to the world of DevOps! Today, we’ll explore how to use **Shell scripting** to tackle a real-world challenge: **reducing infrastructure costs** by efficiently storing Jenkins logs in **AWS S3**, one of the most cost-effective storage services.

Imagine a company like **Google**, running thousands of microservices that constantly generate logs, metrics, and traces. These logs are essential for troubleshooting and observability, and many companies use popular stacks like **ELK** or **EFK**to manage them.

Now, let’s break down the types of logs typically collected:

* **Application Logs**: High-priority logs for debugging application issues.
* **Kubernetes Control-Plane Logs**: High-priority logs for diagnosing cluster problems.
* **Infrastructure Logs**: Logs from tools like Jenkins or Terraform, often used temporarily for troubleshooting build failures.

While application and control-plane logs need to stay accessible, infrastructure logs don’t necessarily need to be stored on expensive VMs. For example, if a Jenkins build fails, developers receive notifications via email or Slack and quickly use the logs to debug. Once the issue is resolved, retaining these logs on servers becomes unnecessary.

However, keeping a backup for **future reference or compliance** is still crucial. That’s where **AWS S3** comes in! By offloading these logs to S3 using a **simple shell script**, we achieve a balance between cost optimisation and reliable backup storage.

**💡 Prerequisites:**

* **An AWS Account**: To store logs in S3, you’ll need access to AWS services. If you don’t have an account, create one at [aws.amazon.com](http://aws.amazon.com/).
* **Basic Understanding of Shell Scripting**: Familiarity with creating and running simple shell scripts will be helpful. Don’t worry; I’ll guide you through the necessary steps!
* **Basic Knowledge of AWS S3**: You should know how to create buckets and manage objects in S3.
* **Jenkins Installed**: Make sure Jenkins is set up on your infrastructure, as we’ll be working with its build logs.

**Setting Up for the Script**

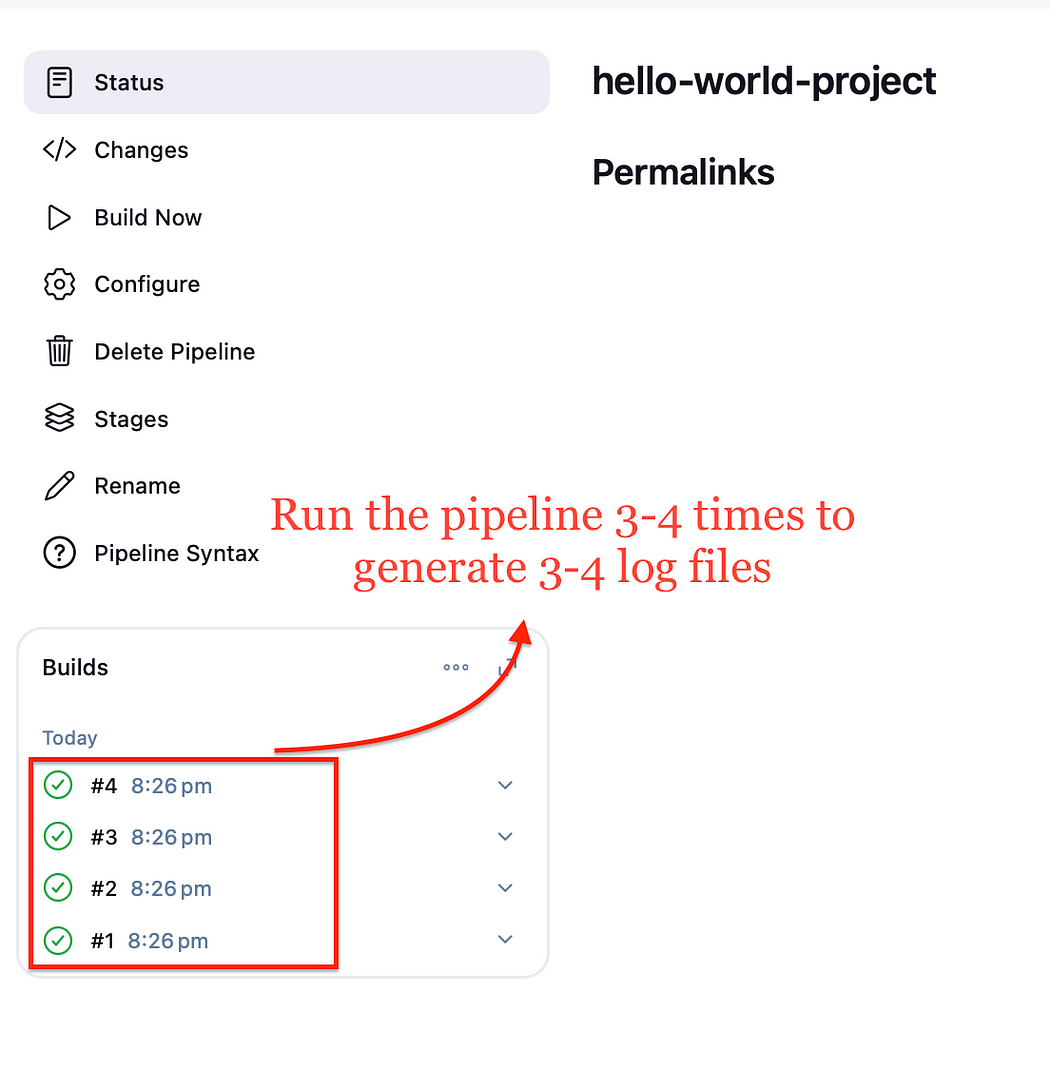
Before diving into the script, let’s prepare everything we need to get started:

1. **Create a Sample Jenkins Pipeline Project**:

* Log in to Jenkins and create a **pipeline project** named hello-world-project.
* Use a basic **Hello World template** for the pipeline.

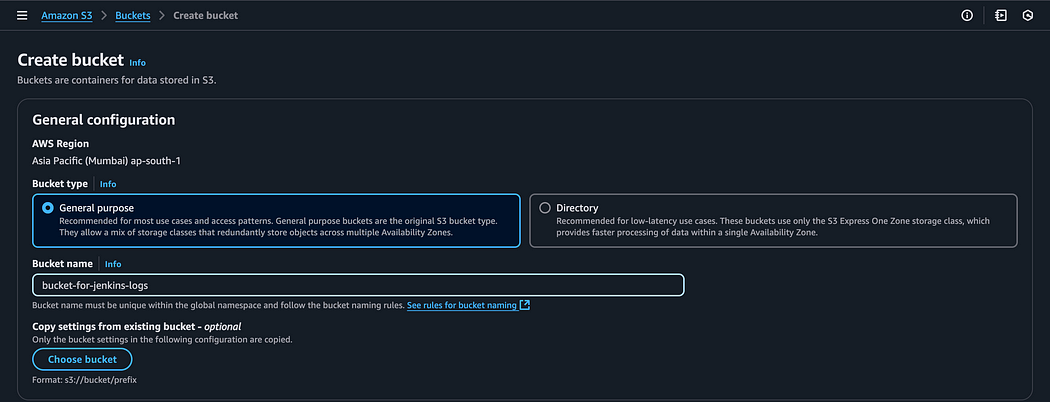


* Run the pipeline 3–4 times to generate a few build log files. These logs will be the ones we push to S3.



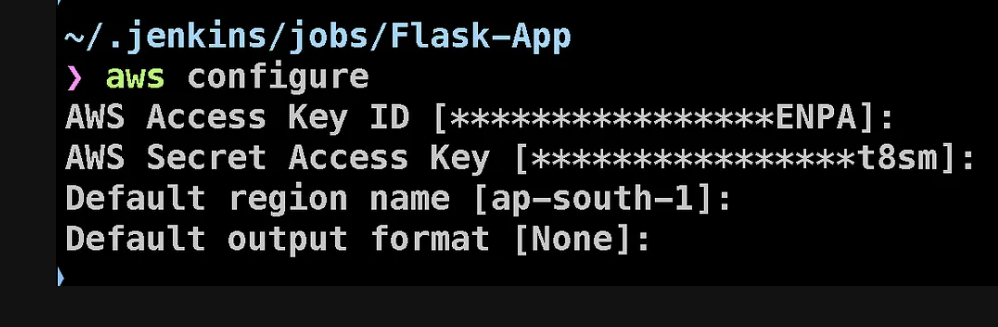
**2. Set Up an S3 Bucket**:

* Head over to your AWS Management Console and create a new S3 bucket.
* Give it a unique name, for example, bucket-for-jenkins-logs (remember, S3 bucket names must be globally unique



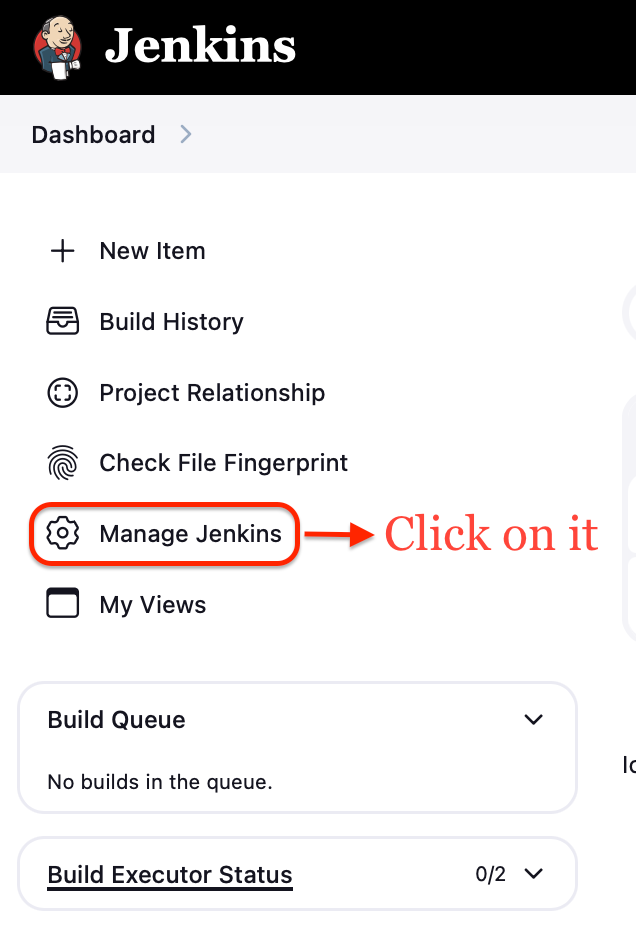
**. Configure the AWS CLI**:

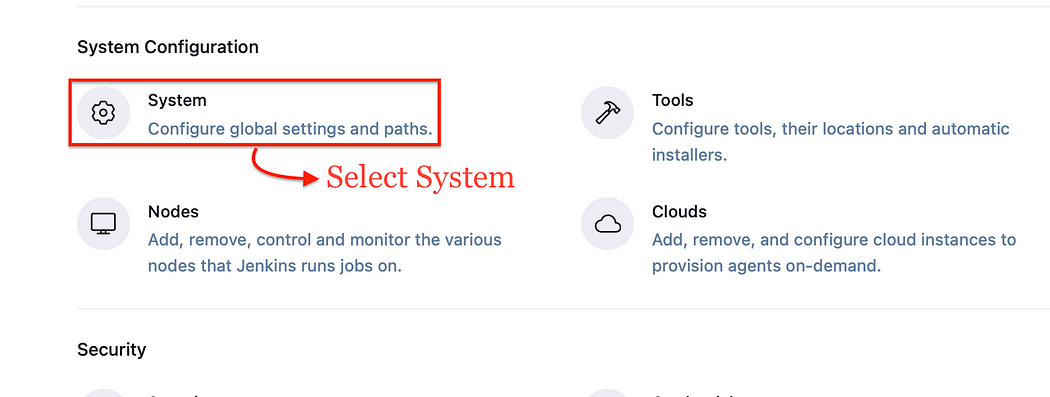
* Generate an **Access Key** and **Secret Key** from the AWS IAM Console.
* Use these credentials to configure your AWS CLI by running the command:

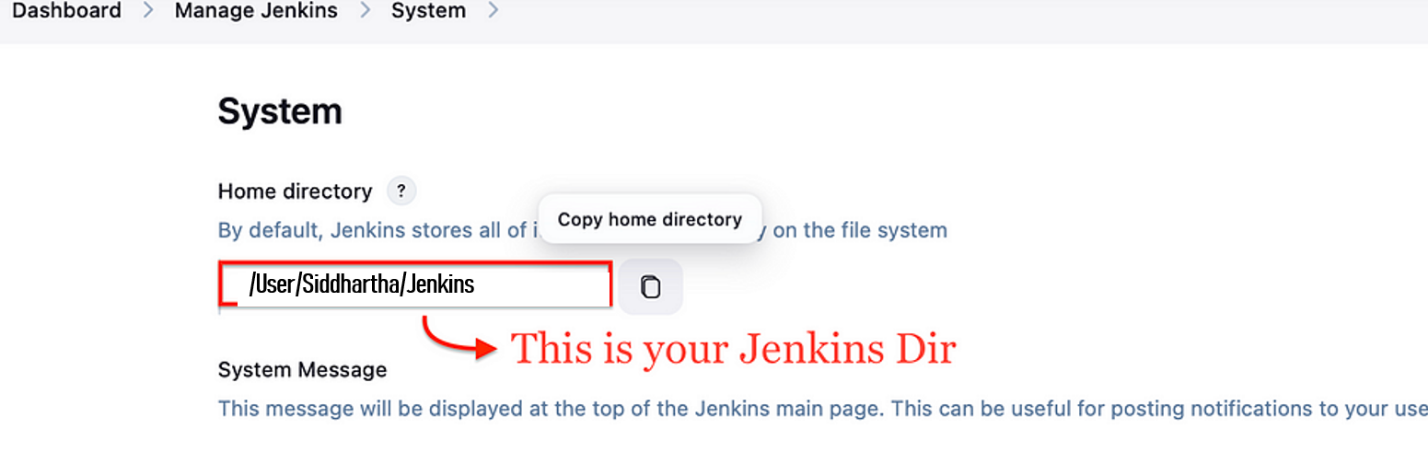


**. Identify Jenkins Home Directory**:

* The script needs to know where your Jenkins home directory is located.
* To find this, navigate to **Manage Jenkins > System Configuration** in the Jenkins UI. At the top of the page, you’ll see the **JENKINS\_HOME** directory path. Make a note of it.



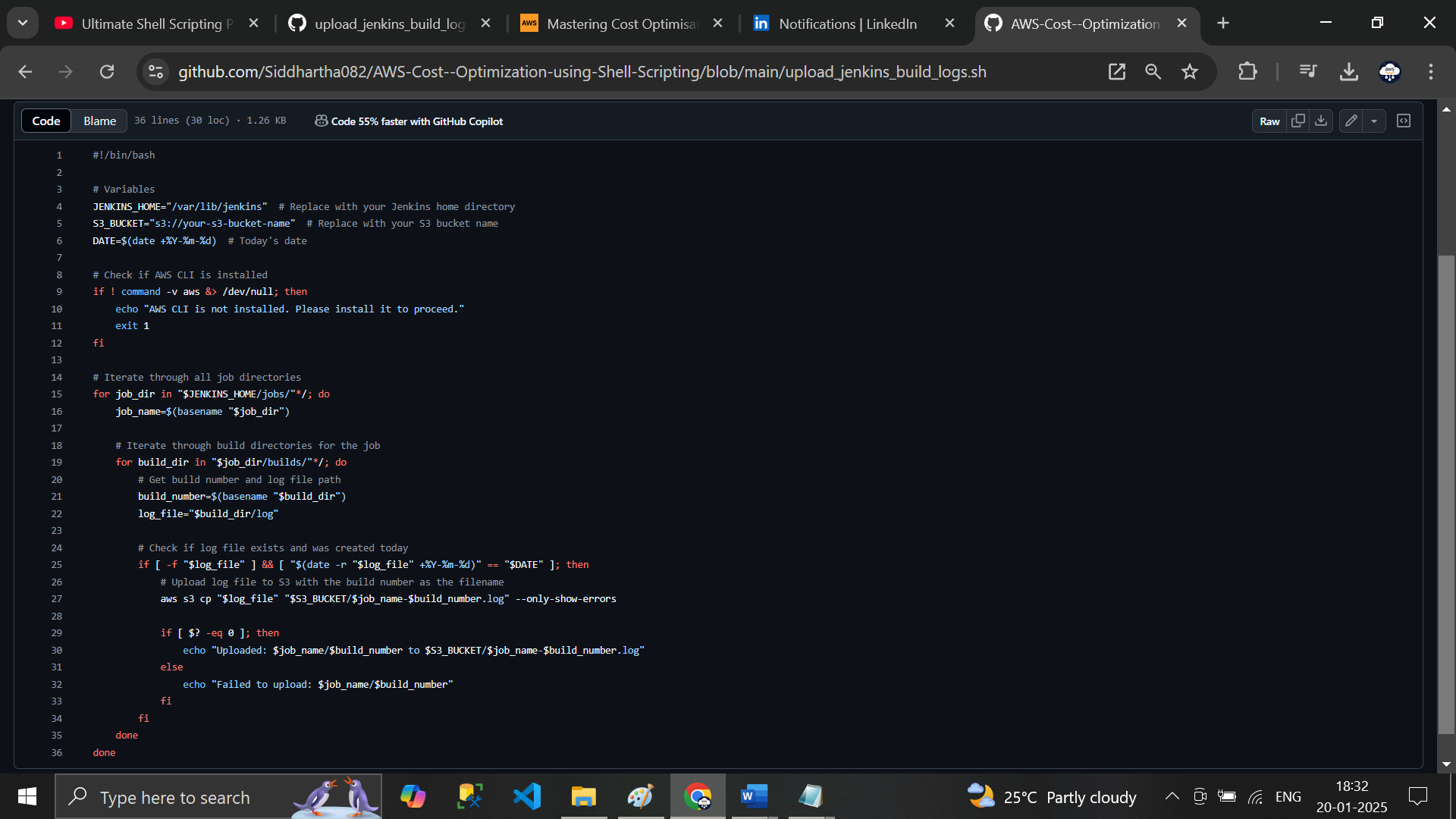


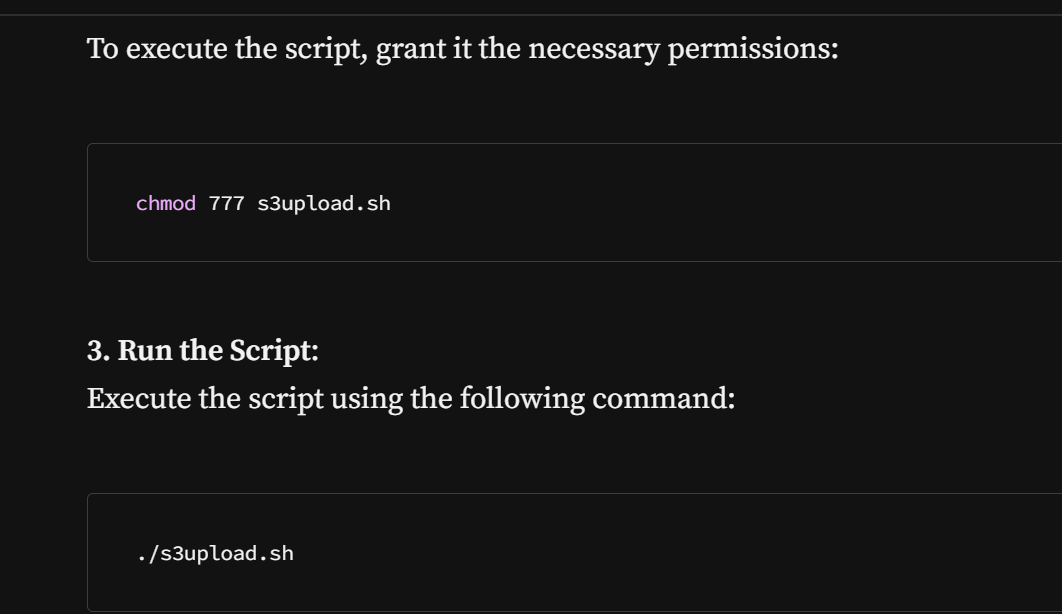


**Writing the Shell Script**

Now, let’s create the script that will handle the automation of uploading Jenkins logs to S3. Follow these steps:

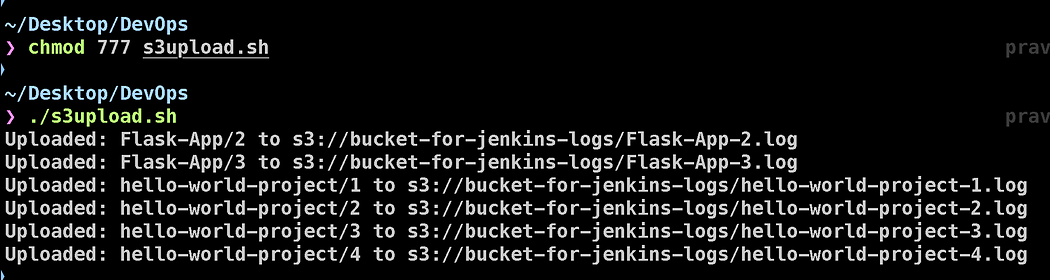
1. **Create the Script File**:  
   Create a file named [s3upload.sh](http://s3upload.sh/) in your preferred directory and add the following content:

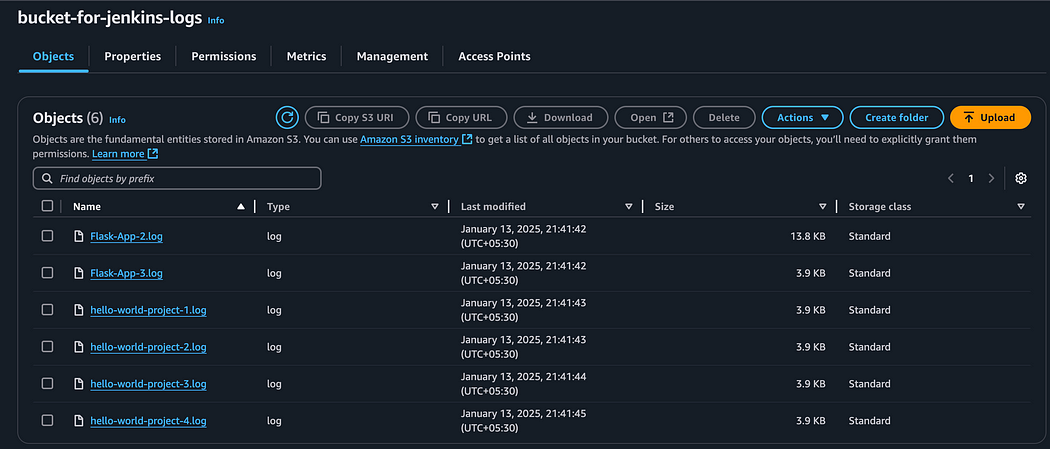




This script will:

* Iterate through the Jenkins jobs and builds directories.
* Check for logs created on the current date.
* Upload the logs to your specified S3 bucket, using the job name and build number as part of the file name.





You will see the logs from your Jenkins dir copied to your s3 bucket.

**Conclusion**

Congratulations! 🎉 You’ve successfully created a shell script that automates the process of uploading Jenkins logs to S3, optimizing storage costs and ensuring easy access to backups. By storing infrastructure logs in S3 instead of VMs, we’ve not only reduced operational costs but also made our log management more efficient and scalable.

But the optimization doesn’t stop here! AWS S3 offers **Lifecycle Management** — a powerful feature that allows you to define rules to automatically transition objects to lower-cost storage classes like **Glacier** or **Deep Archive**. These storage classes are designed for infrequent access, making them perfect for retaining logs that might be needed only for compliance or audits. By leveraging this feature, you can further reduce costs while maintaining long-term access to your logs when required.

This project demonstrates how a simple shell script, combined with cloud services, can solve real-world challenges in a DevOps workflow. It’s a small yet impactful step toward mastering automation and cost optimization in the cloud.