I wanted to build a project that included the following AWS services third party applications:

* Ec2
* S3
* Workspaces
* Security Hub
* Terraform

As such, I used ChatGPT to help me brainstorm some project ideas that I could show case which had solid real-world application. I settled on a Secure File Sharing and Analysis Platform.

**Project Description:**

Create a secure file sharing and analysis platform using AWS services. This platform allows users to upload files securely, which are then stored in S3 buckets. EC2 instances will process these files to analyze their content. AWS Workspaces will provide a virtual desktop environment for users to access and analyze the files further. AWS Security Hub will be used to monitor the security of the entire setup.

**Real-World Application:**

This project can serve as a prototype for a secure document management and analysis system for a healthcare organization. It ensures that sensitive files (e.g., medical records) are securely uploaded, processed, and analyzed while adhering to strict security standards.

**AWS Services Used:**

1. **EC2**: For processing and analyzing the uploaded files.
2. **S3**: For secure storage of the uploaded files.
3. **Workspaces**: For providing a virtual desktop environment for deeper analysis.
4. **Security Hub**: For monitoring and ensuring security compliance.

**Steps to Implement the Project:**

1. **Setup Terraform:**
   * Create a new Terraform configuration file.
2. **Provision S3 Buckets:** 
   * Create an S3 bucket for file uploads.
   * Configure bucket policies and IAM roles to ensure secure access.
3. **Launch EC2 Instances:**
   * Use Terraform to launch EC2 instances for processing files.
   * Write a simple script that the EC2 instance will use to process files from the S3 bucket.
4. **Setup AWS Workspaces:**
   * Use Terraform to provision AWS Workspaces.
   * Ensure that the Workspaces can access the S3 bucket and EC2 instances.
5. **Configure AWS Security Hub:**
   * Enable AWS Security Hub to monitor your AWS environment.
   * Set up necessary compliance checks and alerts.
6. **Write and Deploy Terraform Configuration:**
   * Define the Terraform configurations for all the above components.
   * Deploy the infrastructure using Terraform.
7. **Document the Project:**
   * Write clear documentation on how the system works.
   * Include instructions on how to upload files, how the processing works, and how to access the Workspaces.

**Lessons Learned and Modifications to my original plan:**

1. I originally decided to deploy this section using terraform but determined that probably wouldn’t be the best real-world application for the S3 buckets and security policies. The security policies should be maintained in their own terraform project and this bucket would never be deleted. I left the creation of these resources in the code but commented them out.
2. When I originally implemented the IAM policies I had difficulties getting one of them to correctly translate using the jsonencode() function. I was getting a successful resource creation notification from my Terraform Apply, but the resource was not actually in AWS. This led me to learn about the “EOF” functionality for terraform. I used EOF to create a multiline string of the JSON policy I generated within AWS to import the code directly without translation.

**Improving the project:**

**Hardening the Infrastructure**

* IP Access Controls for Work Spaces – Lock down the Ips that access the workstations.