**CREATING A BULK OF USERS IN AWS IAM FOR A SALES DEPARTMENT**

**Summary**

In this simulated project, I created a bulk of users for the sales department of a company using AWS Identity and Access Management. A policy restricting access to only the required services was assigned to the User group. Using AWS CLI and a couple of bash scripts, I enabled Multi-Factor Authentication for all users in the Sales group.

**Setup/Tools Used**

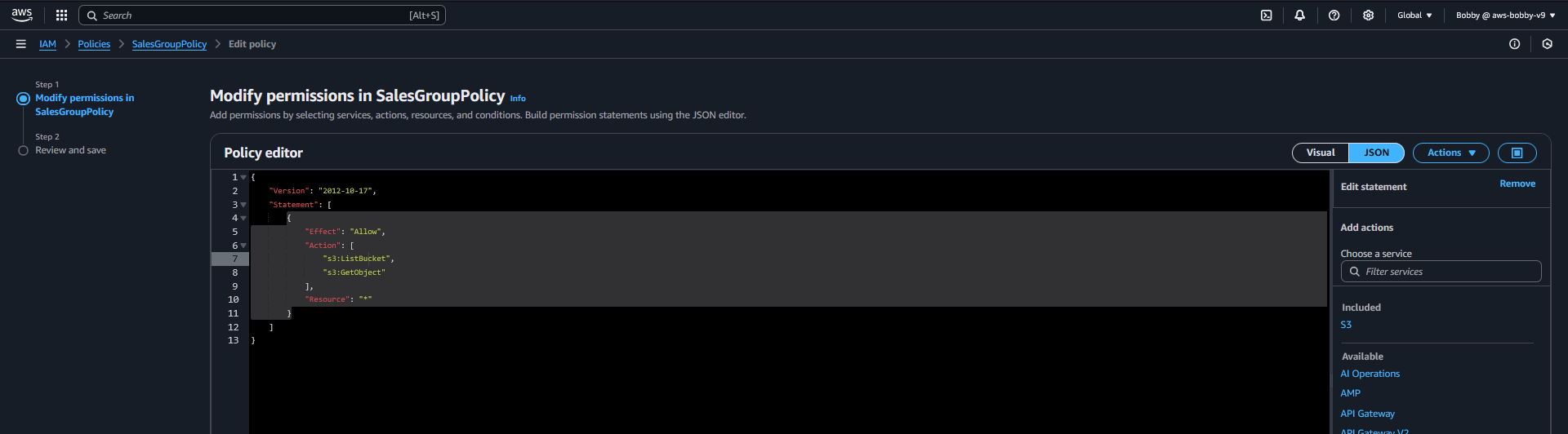
For this project, Visual Studio code IDE was used to program all tasks. The AWS account with full Administrator access permissions as well as all relevant extensions like AWS Toolkit were linked to VScode. On the AWS management console and CLI, we used the IAM service under the administrator account name Bobby. The default operating system used is the WSL Ubuntu extension for Windows on VScode.

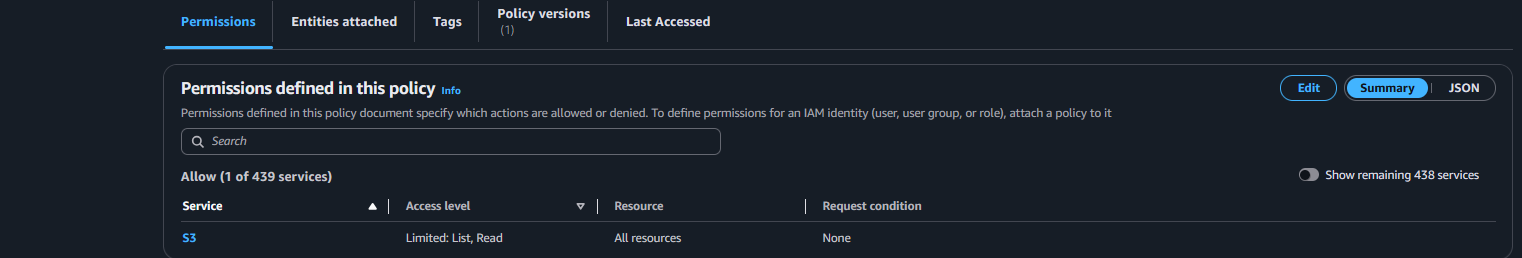
**Task and Procedures**

The first step was to create a user group with a policy conforming to the principle of least privileges. The users that will be created within this group would have access to a limited number of services relevant to their Sales role.

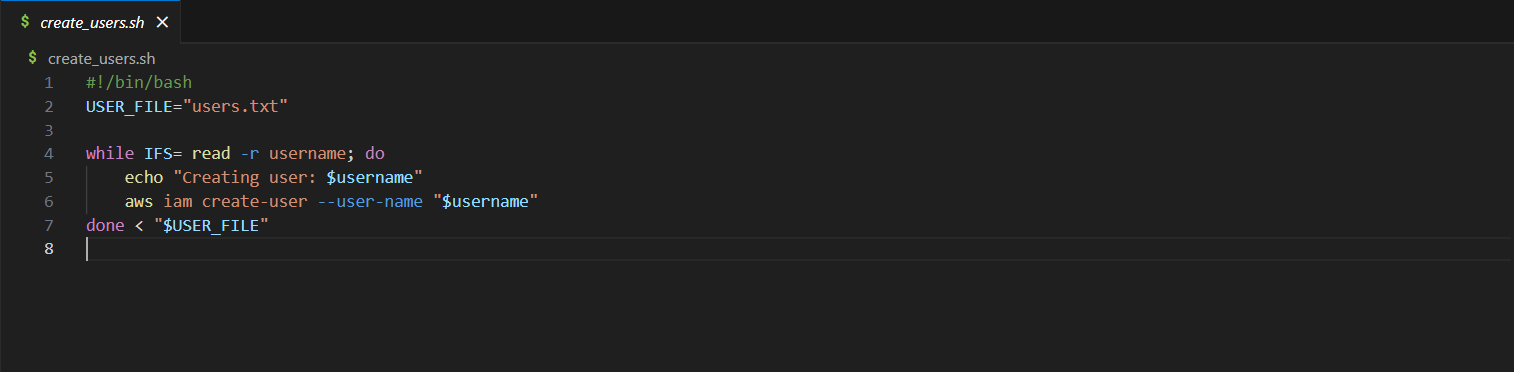
A screenshot of a computer

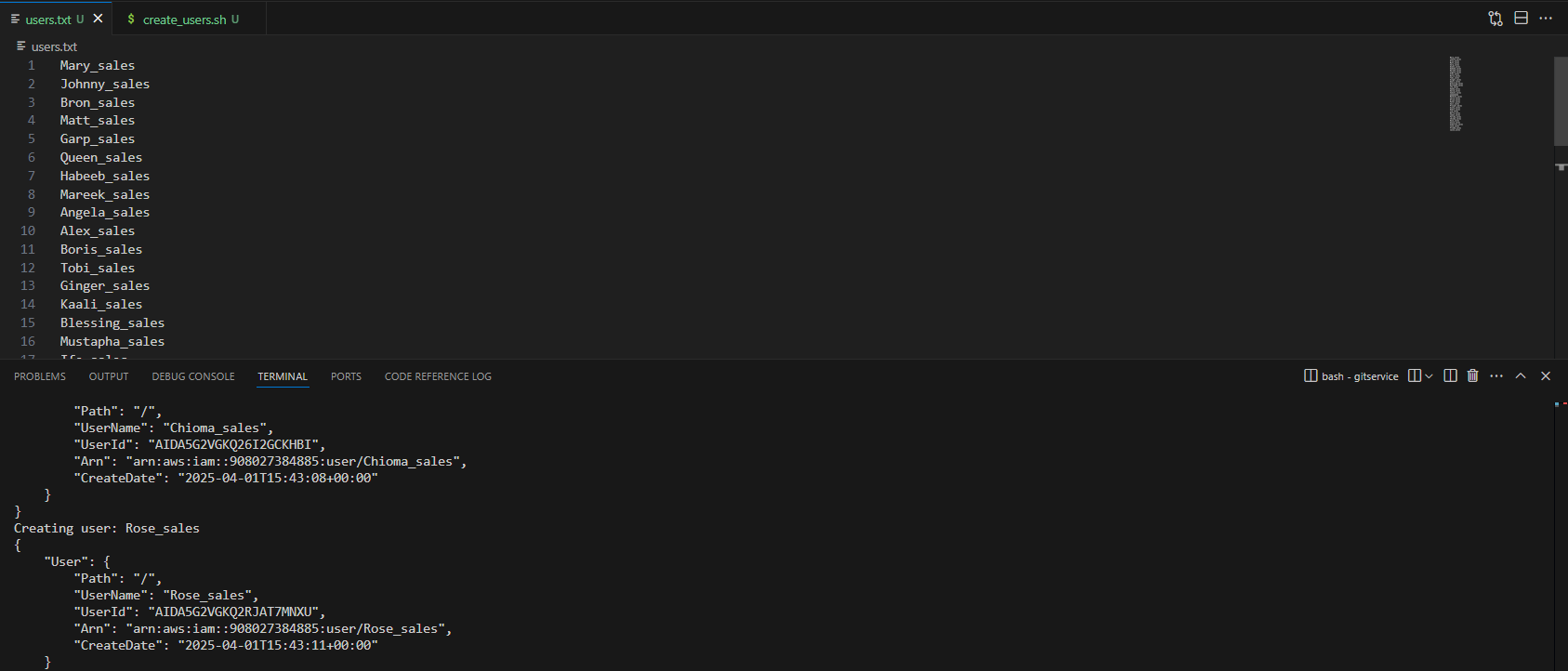
AI-generated content may be incorrect.

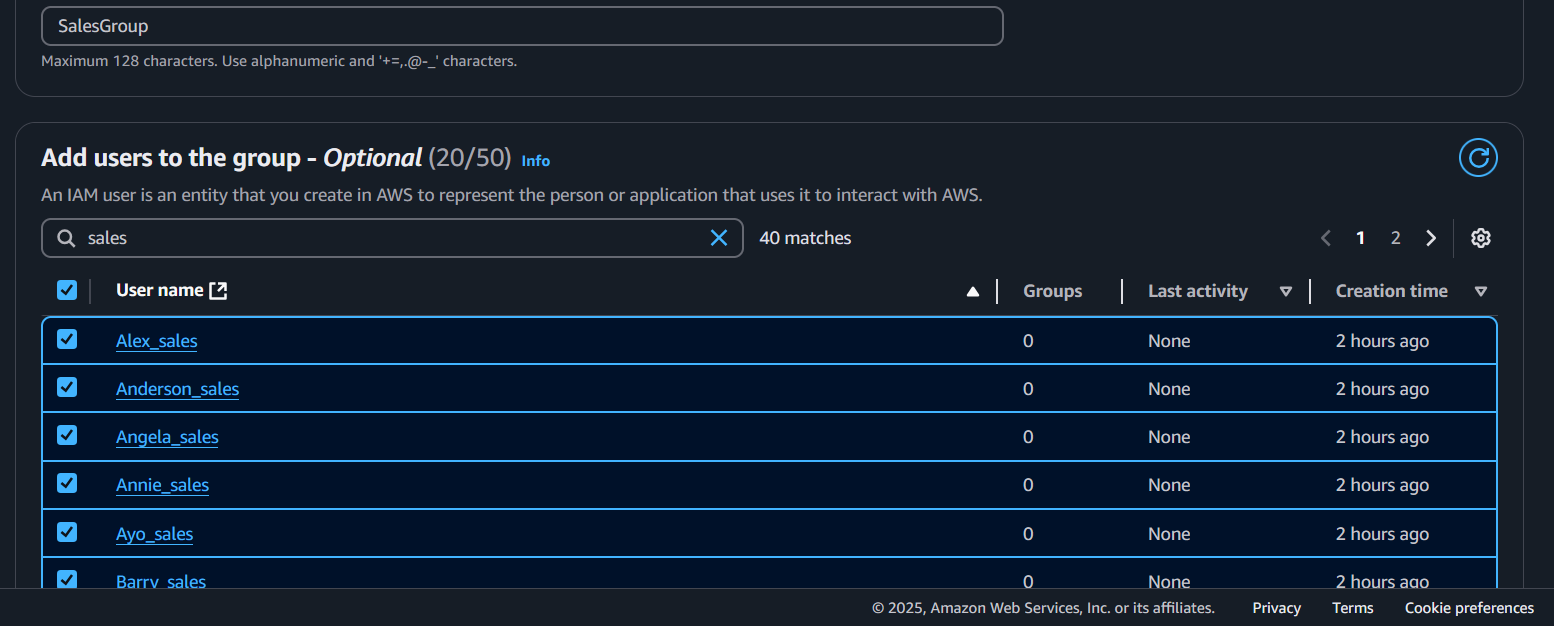




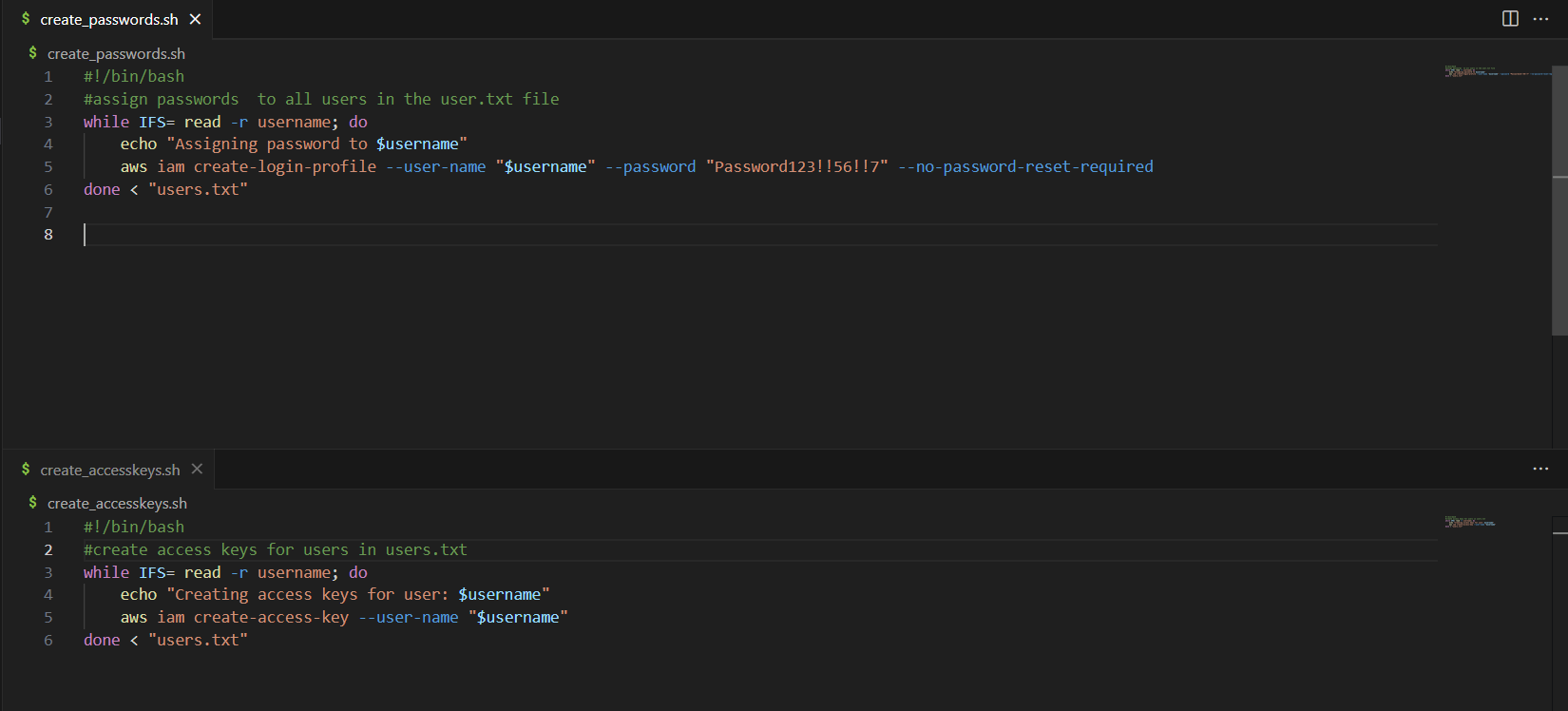
Next Using bash scripting, I created up to 40 users and assigned them to the sales group. The firsts step in this process was to open the terminal in VScode, make sure I was working in the project directory, using touch command, create a **user.txt** file and a **create\_users.sh** file. In the **user.txt** file I wrote down a list of names of all users eg. “Bill\_sales”. In the create\_users.sh file I wrote a script that allows for the creation of the 40 users. Using the command **chmod +x create\_users.sh**, I made the code executable, then ran the script with **./create\_users.sh**. This created all the users listed in the **user.txt** file. I went to the AWS management console IAM users for confirmation and added all the users to the Salesgroup.







After creating users, the next step was to assign passwords and programmatic access to these users. To do this, using the touch command in the terminal, the files **create\_passwords.sh** and **create\_accesskeys.sh** were made.

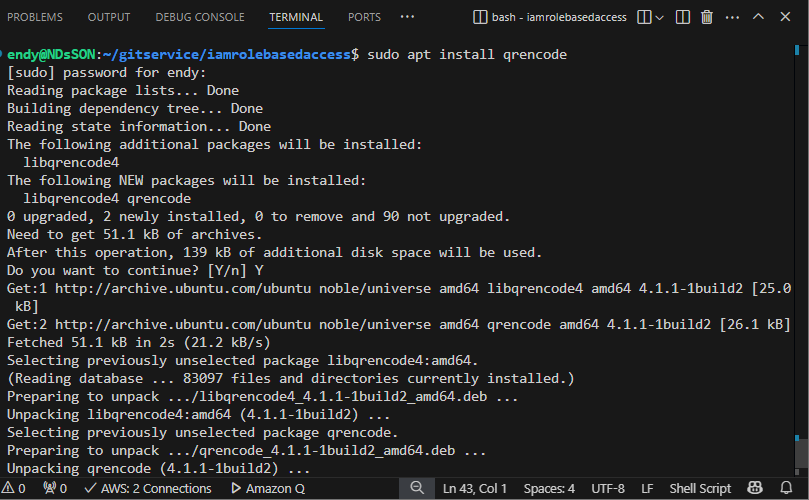


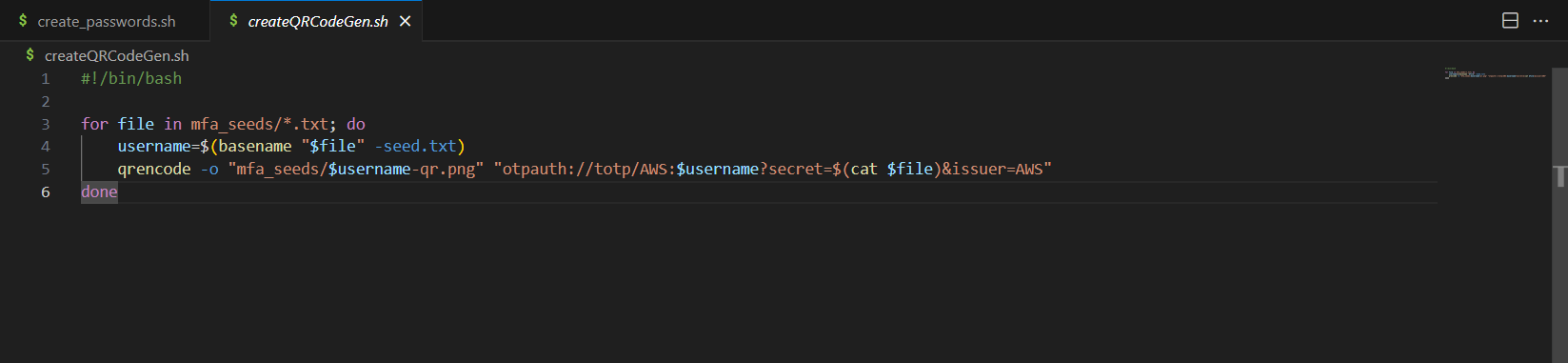
After making these scripts executable and running them. I logged in successfully to one of the sales user accounts to confirm it worked.

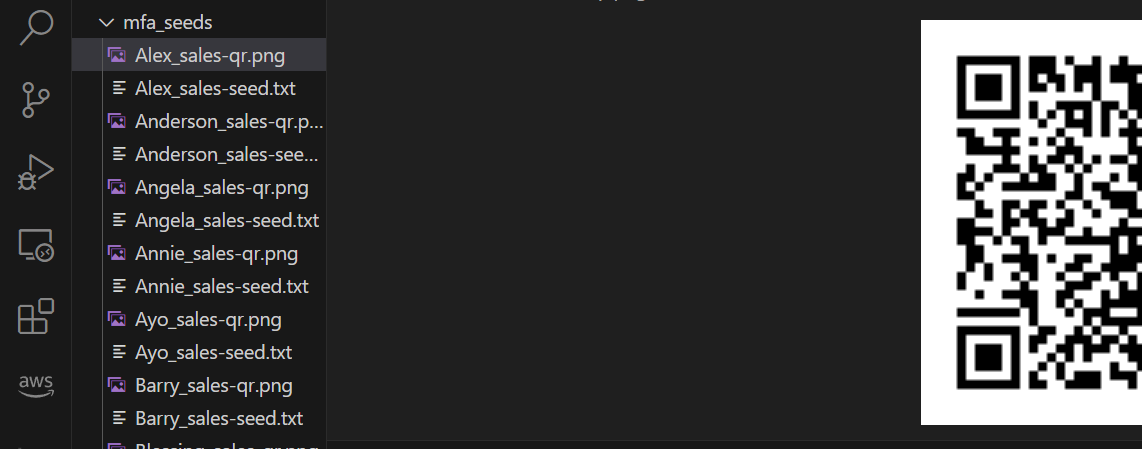
The next step was to add Multi-Factor Authentication to the bulk of users. First, I ensured the admin account had the appropriate permissions to allow for the creation of Virtual Multi-Factor Authentication. Using the AWS management console, we created a Json file with the permissions required.

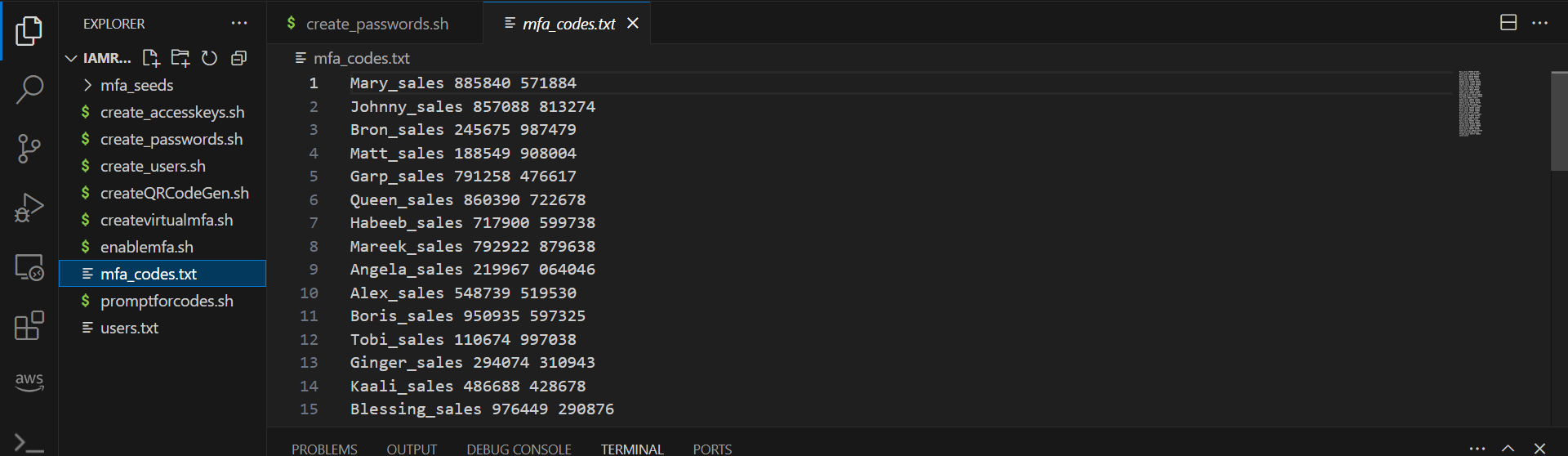


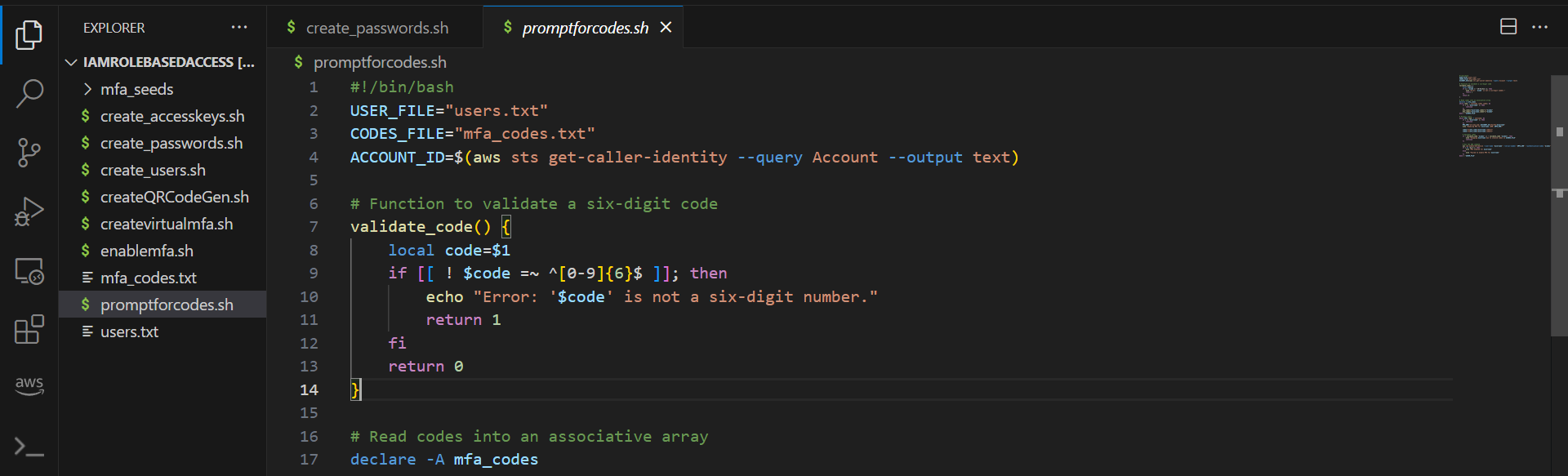
Next, I created a script to enable multi factor authentication using QR codes. First, using the command **sudo apt install qrencode**, we installed the QR-code dependency, then with the script **createQRcodeGen.sh** we created the QR-codes for each user.

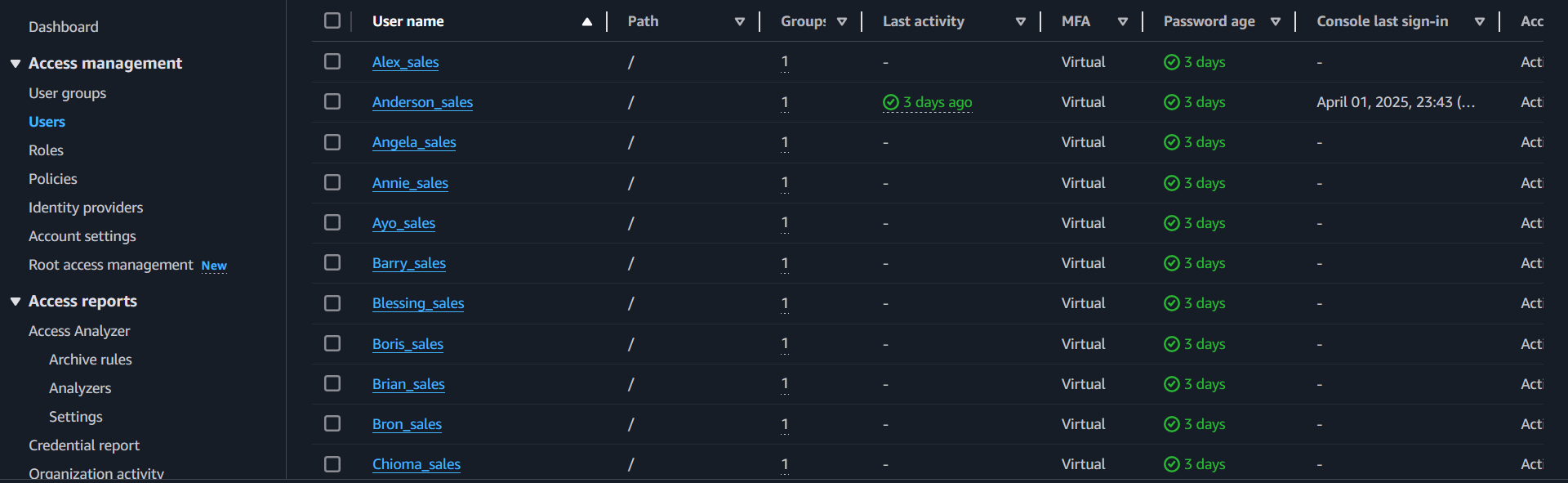






Practically, these QR-codes will be emailed to each SalesGroup user to enable multi-factor authentication on their accounts. But for this project, I used the google authenticator app to record the codes for all users into the **file mfa\_codes.txt**. Then created and executed the script **promptforcodes.sh** to automatically enable mfa for the bulk of users. Then we went to the AWS management console to confirm that the virtual MFA was enabled. 





**Takeaway**

Other methods that I could have used to complete this project is the AWS SDK (Python boto3) or terraform. Ideally, from my recent studies, terraform would be the most efficient way to create this project due to the simplicity of the HashiCorp Language syntax leaving little room for errors and ease of troubleshooting when problems arise.