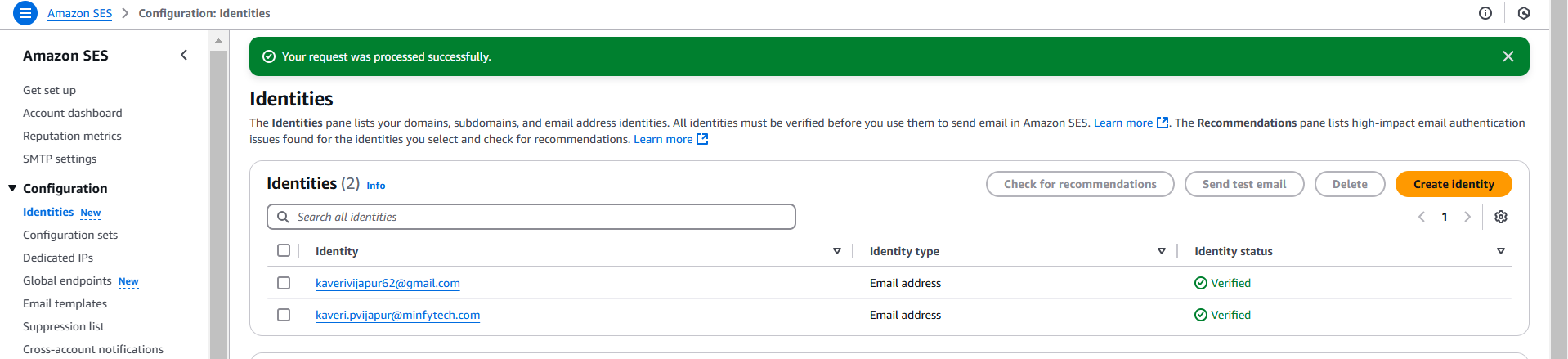
**Applications monitoring automation inside server**

Step1: Install awscli Refer link - <https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>

* Configure awscli
* Create security credentials for console level access > run command > aws config > provide secrete key and access key.

Step2: **Use AWS SES for Sending Emails**

* Set up AWS SES:
* Go to the Amazon SES Console and create a new SMTP credential.
* Once the credentials are created, you will receive a username and password.
* Go to user you created in IAM and attach SESfullaccess policy
* Then go to Identities > create identity – means whitelisting mail ID’s of sender and receiver > create identity.



Step3: Install Python

Commands:

* sudo add-apt-repository universe
* sudo apt update
* sudo apt install python3-pip -y
* pip3 –version

virtual environment to install and manage your Python packages.

**Install python3-venv**

Command:

* sudo apt install python3-venv

**Create a Virtual Environment**

Now, create a new directory for your project and create a virtual environment in it:

Commands:

* mkdir my\_project
* cd my\_project
* python3 -m venv venv

**Activate the Virtual Environment**

Command: source venv/bin/activate

Step5: Python with **pandas** to format the **top** output into an Excel file.

Install Dependencies:

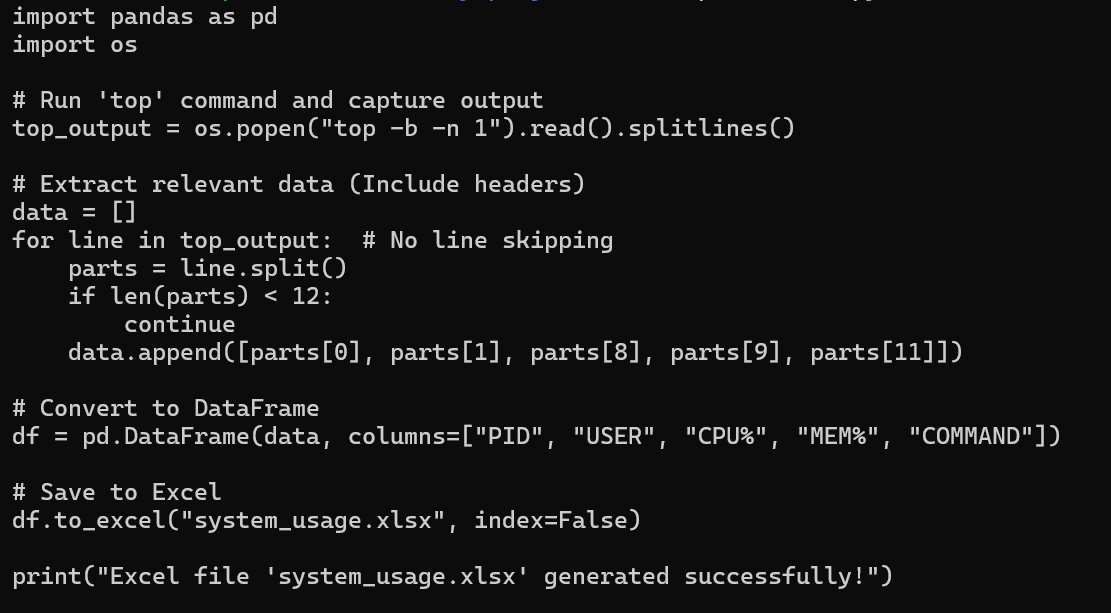
Commands:

* sudo apt install python3-pip -y
* pip3 install pandas openpyxl

Step6: Create a .py file for script – this script

Commands:

* sudo vi top\_to\_excel.py
* add the following script > save > Run the Script > python3 top\_to\_excel.py > Excel file 'system\_usage.xlsx' generated successfully! > verify that system\_usage.xlsx.



This above script runs the top command to capture the current system processes, extracts key information about each process (PID, user, CPU usage, memory usage, and the command), and saves this data to an Excel file named system\_usage.xlsx. The header lines from the top output are ignored, and only valid process data is included in the Excel file.

Step7: Send the File via Email (Using smtplib in Python)

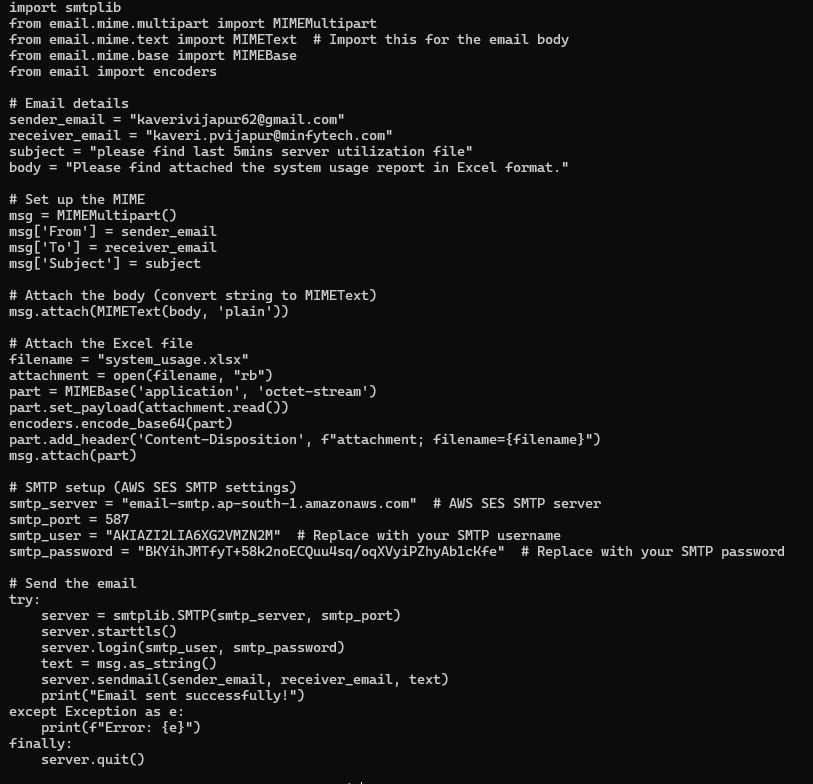
Install required Python libraries smtplib

Command: pip3 install secure-smtplib

automate sending the file via email using Python's smtplib and email modules.

Commands:

* sudo vi mail.py
* open and add following script > save > python3 mail.py



This script sends an email with an attached Excel file (system\_usage.xlsx) using SMTP. It sets up the email details, attaches the file, connects to an SMTP server, logs in, sends the email, and then closes the connection.

**automate the entire Applications Monitoring Automation process using cron jobs**

Step1: Update the crontab to Run Scripts at the Right Intervals

Since top\_to\_excel.py generates system\_usage.xlsx, we need to:

Run top\_to\_excel.py every 3 minutes.

Run mail.py 2 minutes after top\_to\_excel.py to ensure the file is ready.

Edit the Crontab

Run: crontab -e

Then add these lines at the bottom:

\*/3 \* \* \* \* /home/ubuntu/my\_project/venv/bin/python3 /home/ubuntu/my\_project/top\_to\_excel.py

\*/3 \* \* \* \* sleep 120 && /home/ubuntu/my\_project/venv/bin/python3 /home/ubuntu/my\_project/mail.py

Explanation

\*/3 \* \* \* \* → Runs every 3 minutes.

First Line: Runs top\_to\_excel.py to generate the system\_usage.xlsx file.

Second Line: Uses sleep 120 to wait 2 minutes before executing mail.py, ensuring the file is generated before sending the email.