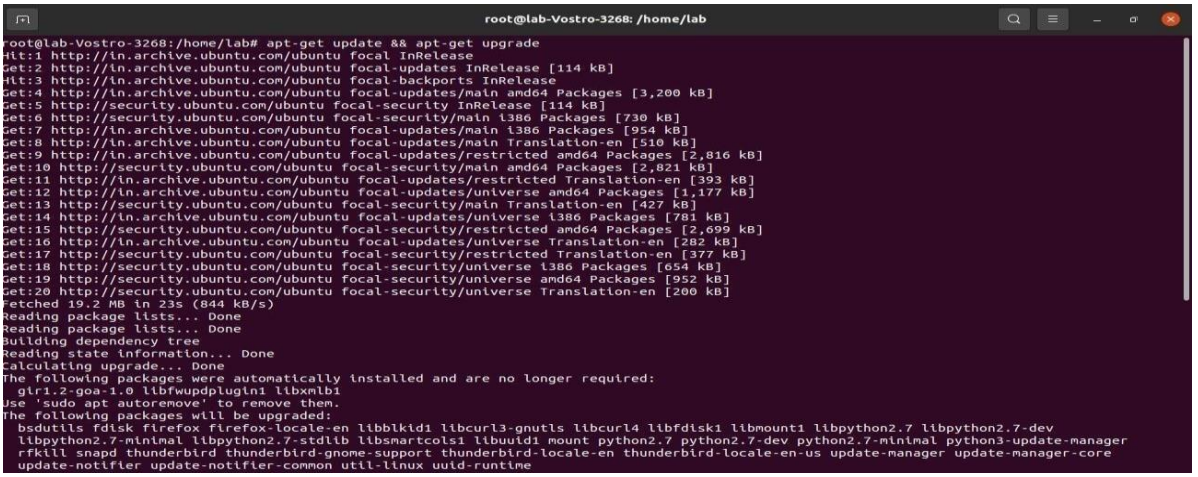
Aim: Using AWS Flow Framework develop application that includes a simple workflow. Workflow calls an activity to print hello world to the console. It must define the basic usage of AWS Flow Framework, including defining contracts, implementation of activities and workflow coordination logic and worker programs to host them.

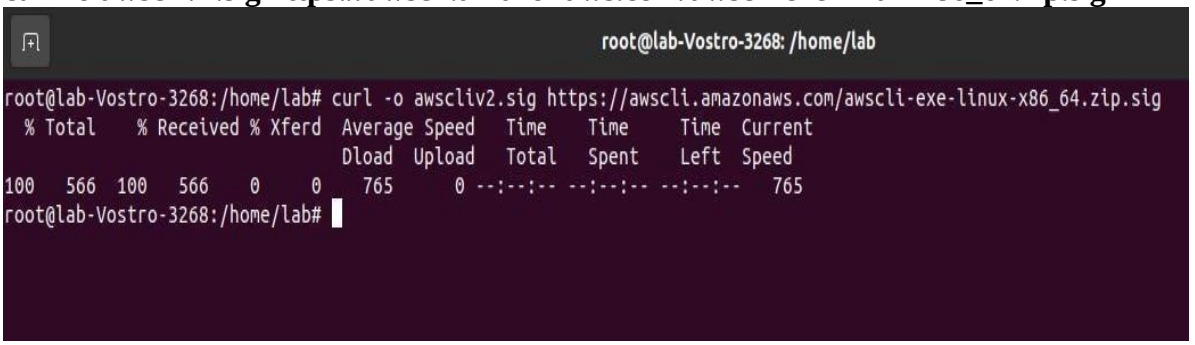
Step 1: Open Terminal and Update and Upgrade your system by command

sudo apt-get update && sudo apt-get upgrade



Step 2: Download awscliv2.zip with command
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"

Step 3: Download awscliv2.sig file with command
curl -o awscliv2.sig https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip.sig



Step 4: unzip awscliv2.zip with command

unzip awscliv2.zip

```
root@lab-Vostro-3268:/home/lab# unzip awscliv2.zip
Archive:  awscliv2.zip
   creating: aws/
   creating: aws/dist/
  inflating: aws/THIRD_PARTY_LICENSES
  inflating: aws/install
  inflating: aws/README.md
   creating: aws/dist/awscli/
   creating: aws/dist/cryptography/
   creating: aws/dist/docutils/
   creating: aws/dist/lib-dynload/
  inflating: aws/dist/aws
  inflating: aws/dist/aws_completer
  inflating: aws/dist/libpython3.11.so.1.0
  inflating: aws/dist/_awscrt.abi3.so
  inflating: aws/dist/_cffi_backend.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/_ruamel_yaml.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/libz.so.1
  inflating: aws/dist/liblzma.so.0
  inflating: aws/dist/libbz2.so.1
  inflating: aws/dist/libffi.so.5
  inflating: aws/dist/libsqlite3.so.0
  inflating: aws/dist/base_library.zip
  inflating: aws/dist/lib-dynload/_pickle.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/lib-dynload/_hashlib.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/lib-dynload/_sha3.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/lib-dynload/_blake2.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/lib-dynload/_sha256.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/lib-dynload/_md5.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/lib-dynload/_sha1.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/lib-dynload/_sha512.cpython-311-x86_64-linux-gnu.so
```

Step 5: Run command
sudo ./aws/install

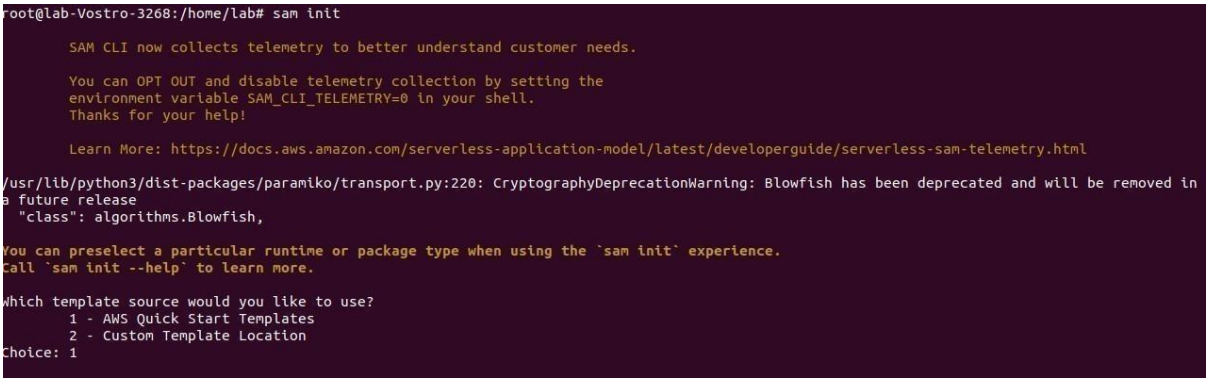
```
root@lab-Vostro-3268:/home/lab#  sudo ./aws/install
You can now run: /usr/local/bin/aws --version
```

Step 6: Type
command pip3
install aws-sam-cli

```
root@lab-Vostro-3268:/home/lab# pip3 install aws-sam-cli
Collecting aws-sam-cli
  Downloading aws_sam_cli-1.113.0-py3-none-any.whl (5.9 MB)
     |████████████████████████████████| 5.9 MB 21 kB/s
Collecting aws-lambda-builders==1.47.0
  Downloading aws_lambda_builders-1.47.0-py3-none-any.whl (130 kB)
     |████████████████████████████████| 130 kB 547 kB/s
Collecting tzlocal==5.2
  Downloading tzlocal-5.2-py3-none-any.whl (17 kB)
Collecting dateparser~=1.2
  Downloading dateparser-1.2.0-py2.py3-none-any.whl (294 kB)
     |████████████████████████████████| 294 kB 2.4 MB/s
Collecting Flask<3.1
  Downloading flask-3.0.2-py3-none-any.whl (101 kB)
     |████████████████████████████████| 101 kB 447 kB/s
Collecting boto3<2,>=1.29.2
  Downloading boto3-1.34.76-py3-none-any.whl (139 kB)
     |████████████████████████████████| 139 kB 558 kB/s
Collecting pyopenssl~=24.1.0
  Downloading pyOpenSSL-24.1.0-py3-none-any.whl (56 kB)
     |████████████████████████████████| 56 kB 580 kB/s
Collecting requests~=2.31.0
  Using cached requests-2.31.0-py3-none-any.whl (62 kB)
Collecting click~=8.1
  Downloading click-8.1.7-py3-none-any.whl (97 kB)
     |████████████████████████████████| 97 kB 377 kB/s
Collecting watchdog==4.0.0
  Downloading watchdog-4.0.0-py3-none-manylinux2014_x86_64.whl (82 kB)
     |████████████████████████████████| 82 kB 113 kB/s
Collecting chevron~=0.12
  Downloading chevron-0.14.0-py3-none-any.whl (11 kB)
Collecting ruamel-yaml~=0.18.6
  Downloading ruamel.yaml-0.18.6-py3-none-any.whl (117 kB)
     |████████████████████████████████| 117 kB 363 kB/s
Collecting aws-sam-translator==1.86.0
```

Step 7: Type command sam init in terminal to launch Sam

CLISelect 1st option to use AWS Quick Ttart Templates



Step 8: Select Template no.1 Hello World Example

Step 9: Type “N” if it ask to use most popular runtime and package type

Open new terminal by pressing ctrl+shift+T and check for python version by command python –version

Select the option according to your python version in my case its option 19- python 3.11

```
Use the most popular runtime and package type? (Python and zip) [y/N]: n

Which runtime would you like to use?
        1 - aot.dotnet7 (provided.al2)
        2 - dotnet8
        3 - dotnet6
        4 - go1.x
        5 - go (provided.al2)
        6 - go (provided.al2023)
        7 - graalvm.java11 (provided.al2)
        8 - graalvm.java17 (provided.al2)
        9 - java21
        10 - java17
        11 - java11
        12 - java8.al2
        13 - nodejs20.x
        14 - nodejs18.x
        15 - nodejs16.x
        16 - python3.9
        17 - python3.8
        18 - python3.12
        19 - python3.11
        20 - python3.10
        21 - ruby3.2
        22 - rust (provided.al2)
        23 - rust (provided.al2023)
Runtime: 17
```

Step 10: Select package type as <u>Zip</u>

```
What package type would you like to use?
        1 - Zip
        2 - Image
Package type: 1
```

Step 11: Now choose Yes option everytime it ask .

Give project name as per your preference in my case its <u>sam-app-test</u>

```
Based on your selections, the only dependency manager available is pip.
We will proceed copying the template using pip.

Would you like to enable X-Ray tracing on the function(s) in your application?  [y/N]: y
X-Ray will incur an additional cost. View https://aws.amazon.com/xray/pricing/ for more details

Would you like to enable monitoring using CloudWatch Application Insights?
For more info, please view https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/cloudwatch-application-insights.html [y/N]: y
AppInsights monitoring may incur additional cost. View https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/appinsights-what-is.html
#appinsights-pricing for more details

Would you like to set Structured Logging in JSON format on your Lambda functions?  [y/N]: y
Structured Logging in JSON format might incur an additional cost. View https://docs.aws.amazon.com/lambda/latest/dg/monitoring-cloudwatchlogs.
html#monitoring-cloudwatchlogs-pricing for more details

Project name [san-app]: san-app-test

Cloning from https://github.com/aws/aws-sam-cli-app-templates (process may take a moment)

    -----------------------
    Generating application:
    -----------------------
    Name: sam-app-test
    Runtime: python3.8
    Architectures: x86_64
    Dependency Manager: pip
    Application Template: hello-world
    Output Directory: .
    Configuration file: san-app-test/samconfig.toml

    Next steps can be found in the README file at sam-app-test/README.md
```

Step 12: Now one folder will be created by your provided project name go into that folder by command cd (folder name)

After entering the project folder we will invoke the HelloWorldFunction by using commandsam local invoke „HelloWorldFunction"

```
root@lab-Vostro-3268:/home/lab/sam-app-test# sam local invoke 'HelloWorldFunction'
/usr/lib/python3/dist-packages/paramiko/transport.py:220: CryptographyDeprecationWarning: Blowfish has been deprecated and will be removed in
a future release
  "class": algorithms.Blowfish,
Invoking app.lambda_handler (python3.8)
Local image was not found.
Removing rapid images for repo public.ecr.aws/sam/emulation-python3.8
Building image.....
.....
.....
.....
Using local image: public.ecr.aws/lambda/python:3.8-rapid-x86_64.

Mounting /home/lab/sam-app-test/hello_world as /var/task:ro,delegated, inside runtime container
START RequestId: 065fefd3-5d41-4b87-bb31-8d820fb635e6 Version: $LATEST
END RequestId: 4b9baa5c-2523-443d-b340-f86e2b139f6a
REPORT RequestId: 4b9baa5c-2523-443d-b340-f86e2b139f6a  Init Duration: 0.06 ms  Duration: 99.11 ms      Billed Duration: 100 ms Memory Size: 1
28 MB   Max Memory Used: 128 MB
{"statusCode": 200, "body": "{\"message\": \"hello world\"}"}
root@lab-Vostro-3268:/home/lab/sam-app-test#
```

It should give you StatusCode:200

Use command sudo snap install docker if docker error occurs.

Step 13: Type command sam local start-api this will give you URL open it in any browser.

```
root@lab-Vostro-3268:/home/lab/sam-app-test# sam local start-api
/usr/lib/python3/dist-packages/paramiko/transport.py:220: CryptographyDeprecationWarning: Blowfish has been deprecated and will be removed in
a future release
  "class": algorithms.Blowfish,
Initializing the lambda functions containers.
Local image was not found.
Removing rapid images for repo public.ecr.aws/sam/emulation-python3.11
Building image.....
.....
Using local image: public.ecr.aws/lambda/python:3.11-rapid-x86_64.

Mounting /home/lab/sam-app-test/hello_world as /var/task:ro,delegated, inside runtime container
Containers Initialization is done.
Mounting HelloWorldFunction at http://127.0.0.1:3000/hello [GET]
You can now browse to the above endpoints to invoke your functions. You do not need to restart/reload SAM CLI while working on your functions,
changes will be reflected instantly/automatically. If you used sam build before running local commands, you will need to re-run sam build for
the changes to be picked up. You only need to restart SAM CLI if you update your AWS SAM template
2024-04-03 09:26:18 WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:3000
2024-04-03 09:26:18 Press CTRL+C to quit
Invoking app.lambda_handler (python3.11)
Reuse the created warm container for Lambda function 'HelloWorldFunction'
Lambda function 'HelloWorldFunction' is already running
START RequestId: 0652f01c-89a7-43d9-85fb-3844c8f1a32b Version: $LATEST
END RequestId: b229bc70-c5b6-4bf2-b9a3-97b2cf9340a1
REPORT RequestId: b229bc70-c5b6-4bf2-b9a3-97b2cf9340a1  Init Duration: 0.04 ms  Duration: 1153.99 ms    Billed Duration: 1154 ms        Memory
 Size: 128 MB   Max Memory Used: 128 MB

No Content-Type given. Defaulting to 'application/json'.
2024-04-03 09:26:38 127.0.0.1 - - [03/Apr/2024 09:26:38] "GET /hello HTTP/1.1" 200 -
2024-04-03 09:26:39 127.0.0.1 - - [03/Apr/2024 09:26:39] "GET /favicon.ico HTTP/1.1" 403 -
```

Output: