# Step Functions Implementation (Workshop-Code-3)

We need some lambda functions, so this is how the workflow will proceed:

Process Input Files

Apply Non-Linearity

Compute Cost Part-1

Compute Cost Part-2

Consolidate Cost

Pre-Reqs:

1. AWS CLI installed and configured.
2. You should have Python 3.x installed and should be able to install any other packages/modules if required.
3. Have access to github
4. Prefer that the participant have their own AWS account to work with, if not, they should be able to create IAM roles, create lambda functions, access a Cloud9 environment if required.
5. Installation of Postman browser.

## Procedure

1. First we write the code for all the Lambda functions and the required helpers in a single file.
2. We then create a layer bundle like in the previous function, this layer will not contain any custom code, this layer will only have the numpy package.
3. We create a layer and publish it as before.
4. We create five lambda function deployment packages, here they are:
   1. process\_input\_files
   2. apply\_non\_linearity
   3. compute\_cost\_1
   4. compute\_cost\_2
   5. consolidate\_cost
   6. initiator

Code for all of the above functions can be found in the lambda\_function.py. Use this to create your deployment packages.

1. Create the functions, make sure you set the memsize for the functions at 192 instead of the standard 128. This may be required as in some cases you may run out of memory with numpy. Refer to the [AWS CLI documentation](https://docs.aws.amazon.com/cli/latest/reference/lambda/index.html) to know how you can set this up during function creation or use the AWS Lambda console to make changes later. Make sure you test your lambda functions like before.
2. Create a role, call it stepfunction-exec-role, add the following authorizations to it:

CloudWatchLogsFullAccess, AWSXrayFullAccess and AWSLambdaRole, both of these are managed policies. Use this role in the AWS CLI command you use next to create the state machine.

1. You should have a mysteps.asl.json file. This file defines your state machine. Create one like so,

aws stepfunctions create-state-machine --name costcompute --role-arn arn:aws:iam::<***AWS Account number***>:role/RNS-StepFuncRole-To-Invoke-Lambda --definition <file://mysteps.asl.json>

Enable logging for the state machine you created by going to the Step Functions console and choosing the details page (click Edit) and making changes to the logging configuration, choose to log ALL and enable Xray tracing.

1. Go to the Step Function console and click on the state machine you just created. Click on Start execution, you will be asked to provide an input. Provide this,

{

“bucket”: “<some bucket>”,

“key”: “<some path all the way to inputdata.zip file>”

}

1. See the execution all the way through. Make sure everything works.
2. Now, we need to be able to send input to the step function. We will use API Gateway to accomplish this.
   1. Go to the APi Gateway Console.
   2. Click on Create API
   3. Choose HTTP API, click on Build under HTTP API.
   4. Click in Add Integrations, choose Lambda.
   5. Choose your ‘initiator’ lambda function.
   6. Give the API a name, say, teststepfunc, click Next in the bottom right corner.
   7. In the next page, choose the method, in our case choose GET. In the resource path write ‘/execarn’. Click Next.
   8. On the next page, give the stage name as ‘mytest1’.
   9. On the next page, review, scroll down to the bottom of the page and click Create.
   10. You should find an invoke URL in the details page of the APi you just created.
   11. Copy the invoke URL and paste it in Postman, add a ‘/’ to the end of the invoke URL and then append the route you defined when you created the API in Postman, add ‘execarn’. Add the following key-value pairs:

bucket - <name of your bucket>

key - <folder/path to your inputdata.zip file in that bucket>

Ensure you use the GET method.

1. See the returned payload. How can you use the returned data to get the computed cost?