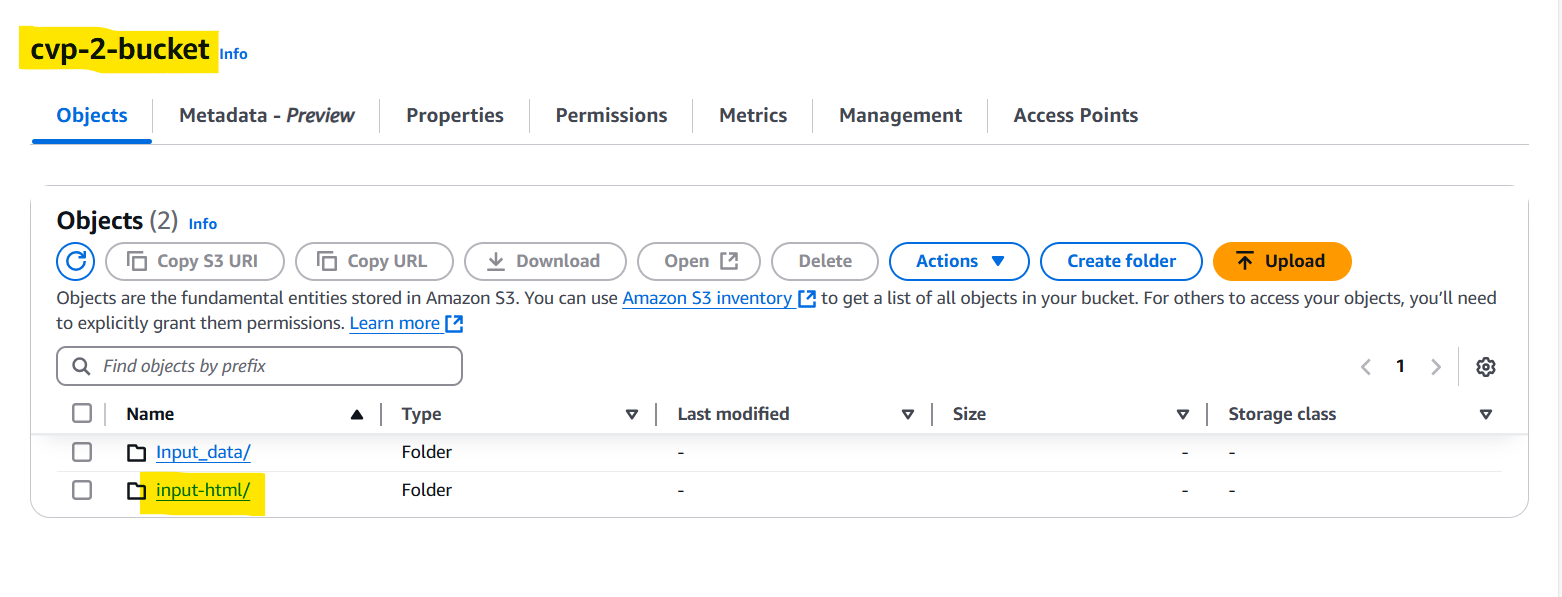
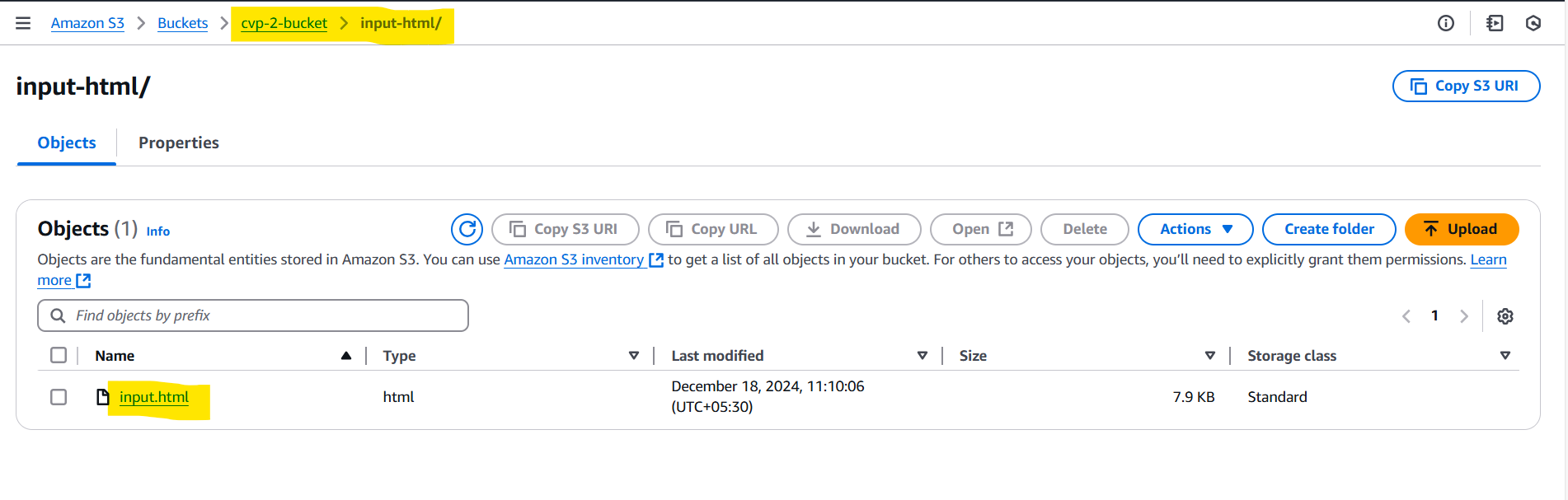
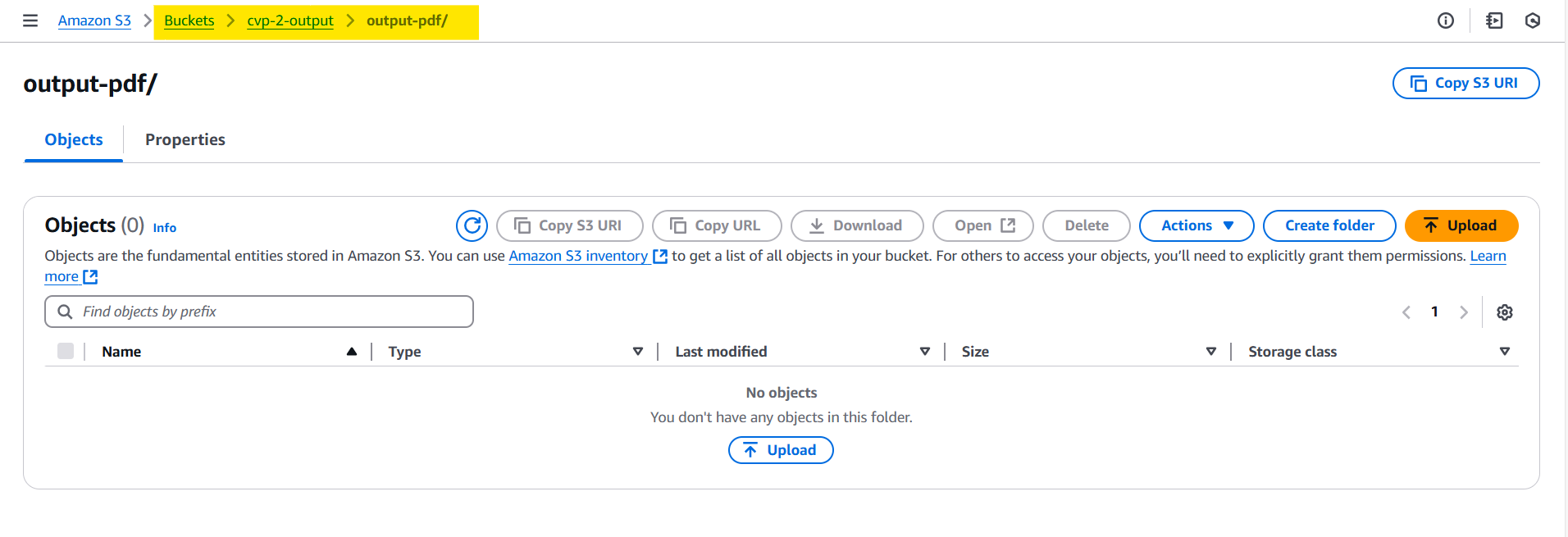
**Convert HTML file to PDF using AWS Lambda in Python Runtime**

1. Context: There is an HTML file in an S3 bucket. We need to convert it into a PDF file and store it in another S3 bucket using AWS Lambda.
2. We are going to use the “pdfkit” library for this and also the “wkhtmltopdf” wrapper, which is a dependency on pdfkit.
3. For this, I created an input bucket called “cvp-2-bucket” and a folder called “input-html” in which I will store my file “input.html”



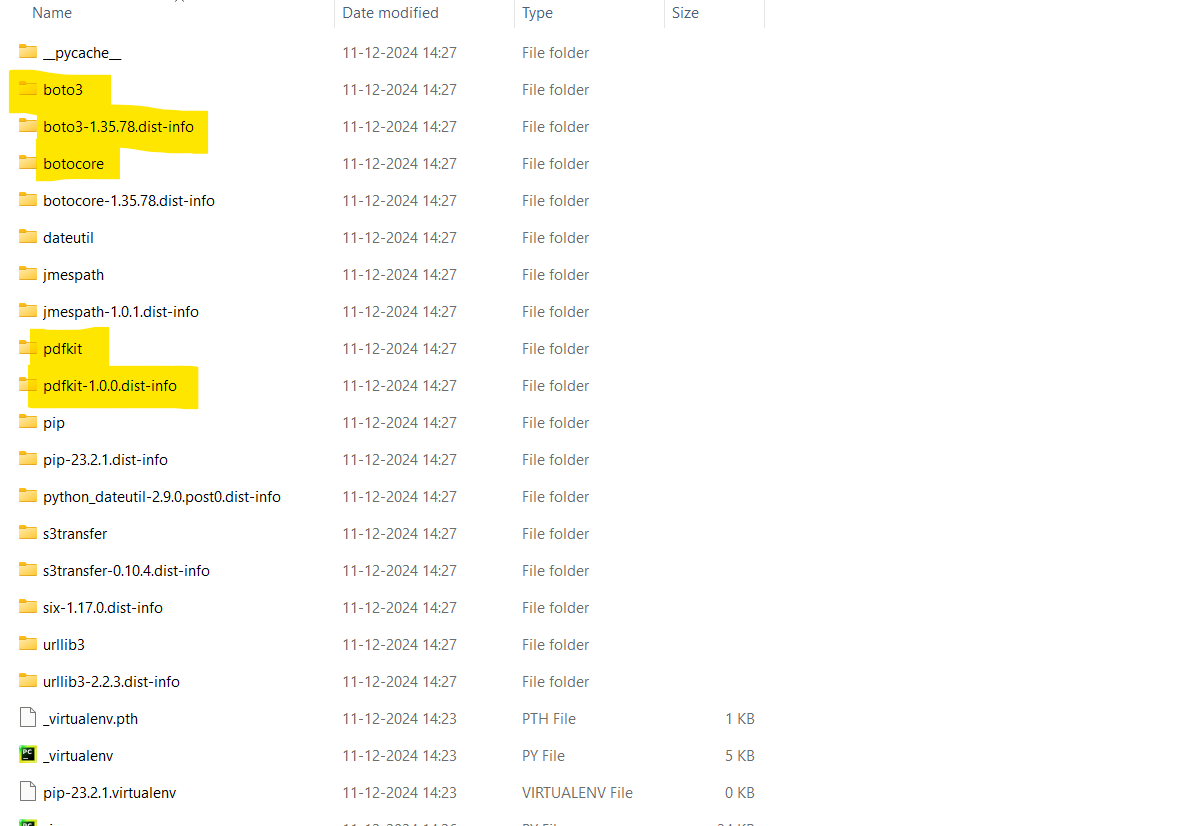


1. Now, I created an output bucket called “cvp-2-output” and a folder called “output-pdf” in which I will store my generated PDF file “output.pdf”

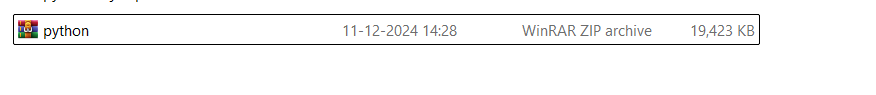


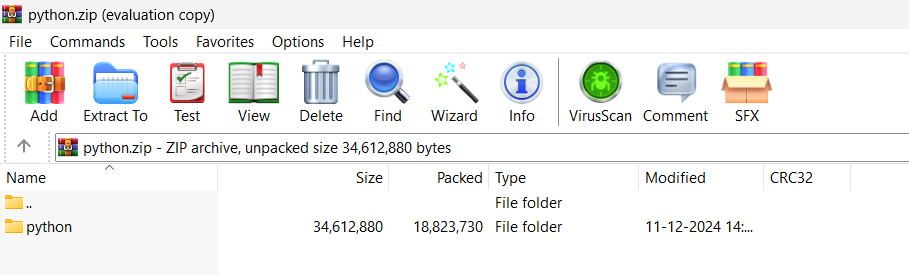
1. Now Create a Lambda Layer:
2. 1st Layer: This consists of the “pdfkit” library and the “boto3” library.

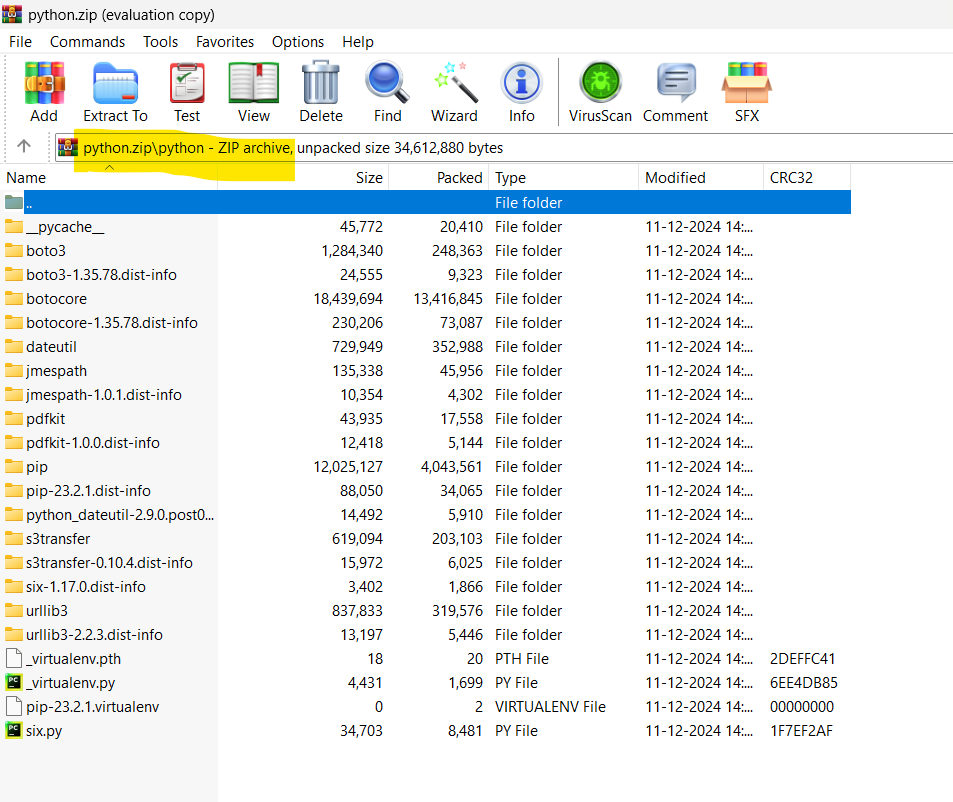
Download that locally and prepare a zip file to be uploaded as a layer.



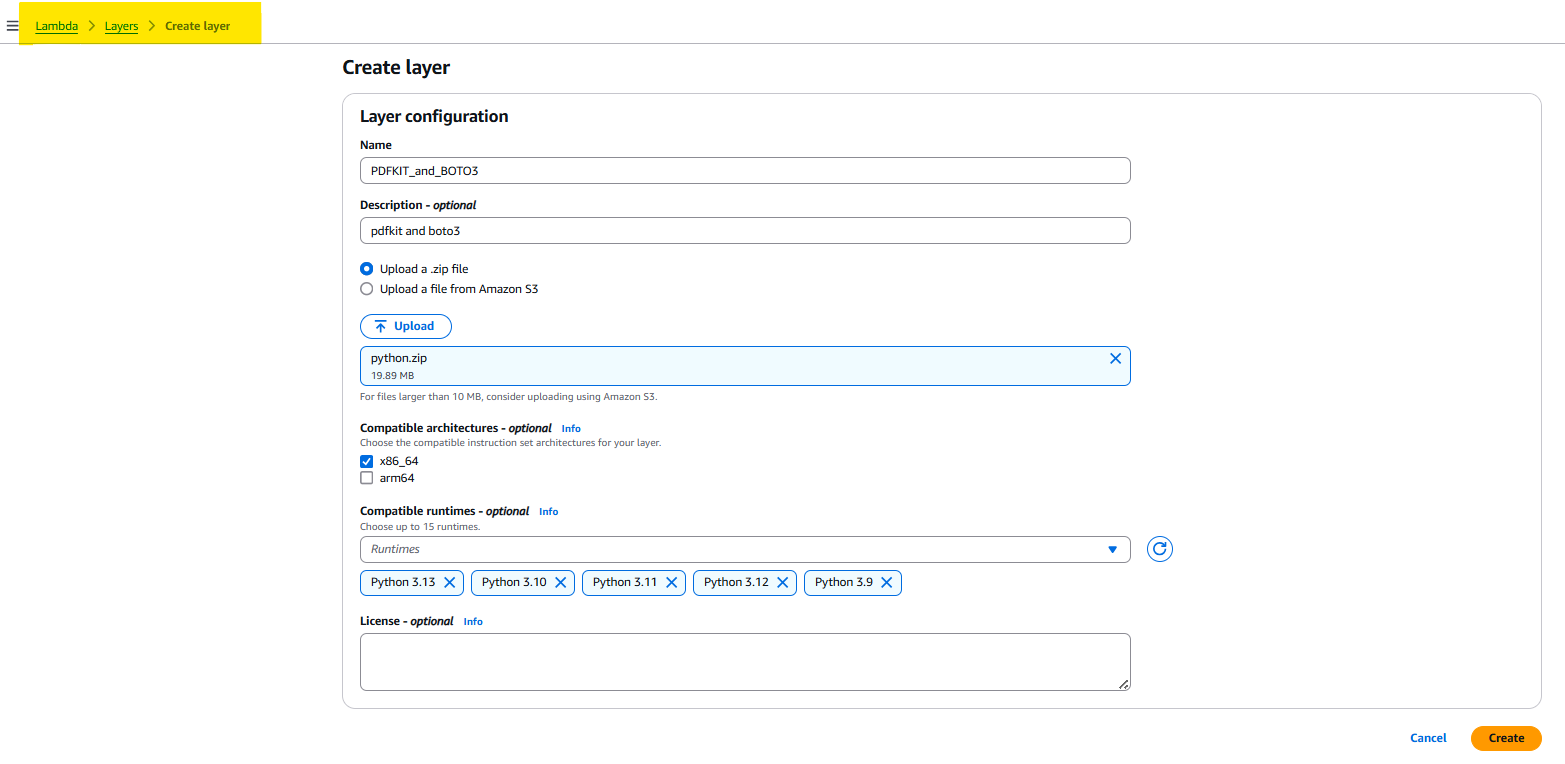
Zip the package under the folder “python” which should look like below







Upload this and create a Lambda layer in AWS as shown below.

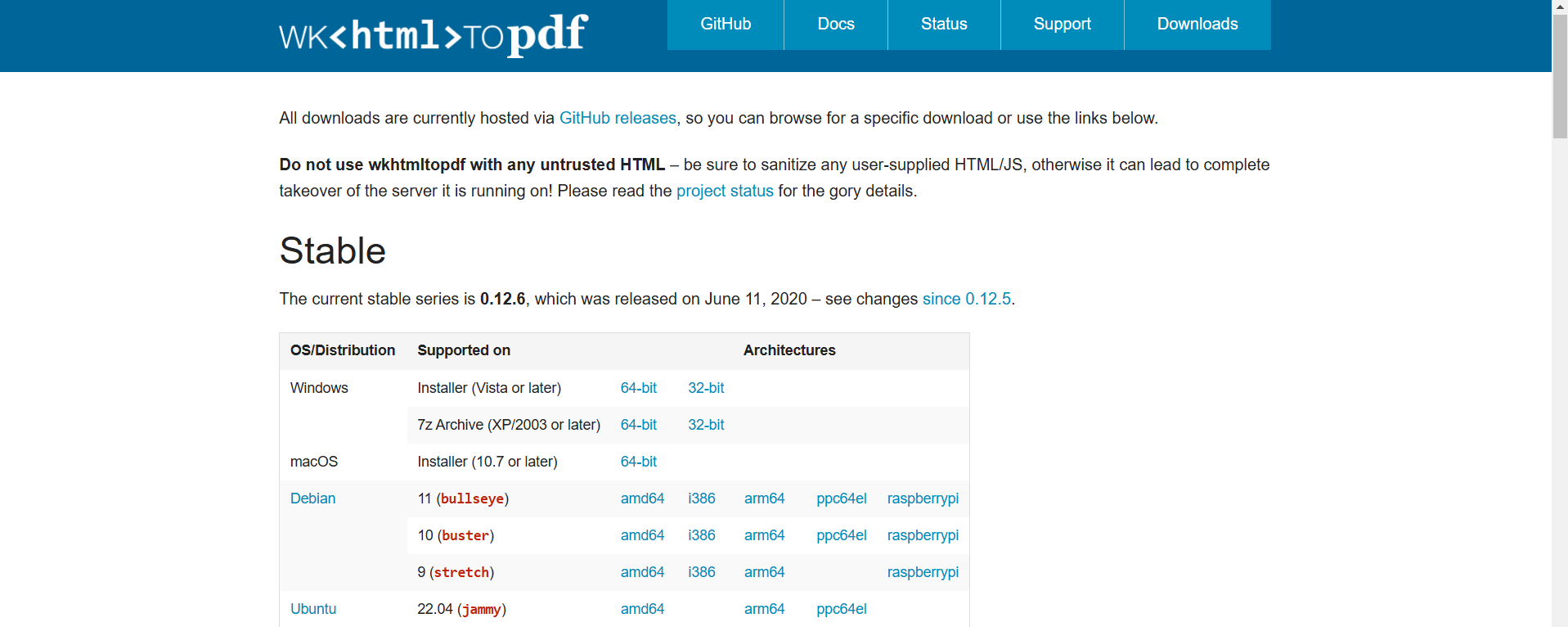


1. 2nd Layer: This consists of the “wkhtmltopdf” wrapper which is a dependency for “pdfkit”.

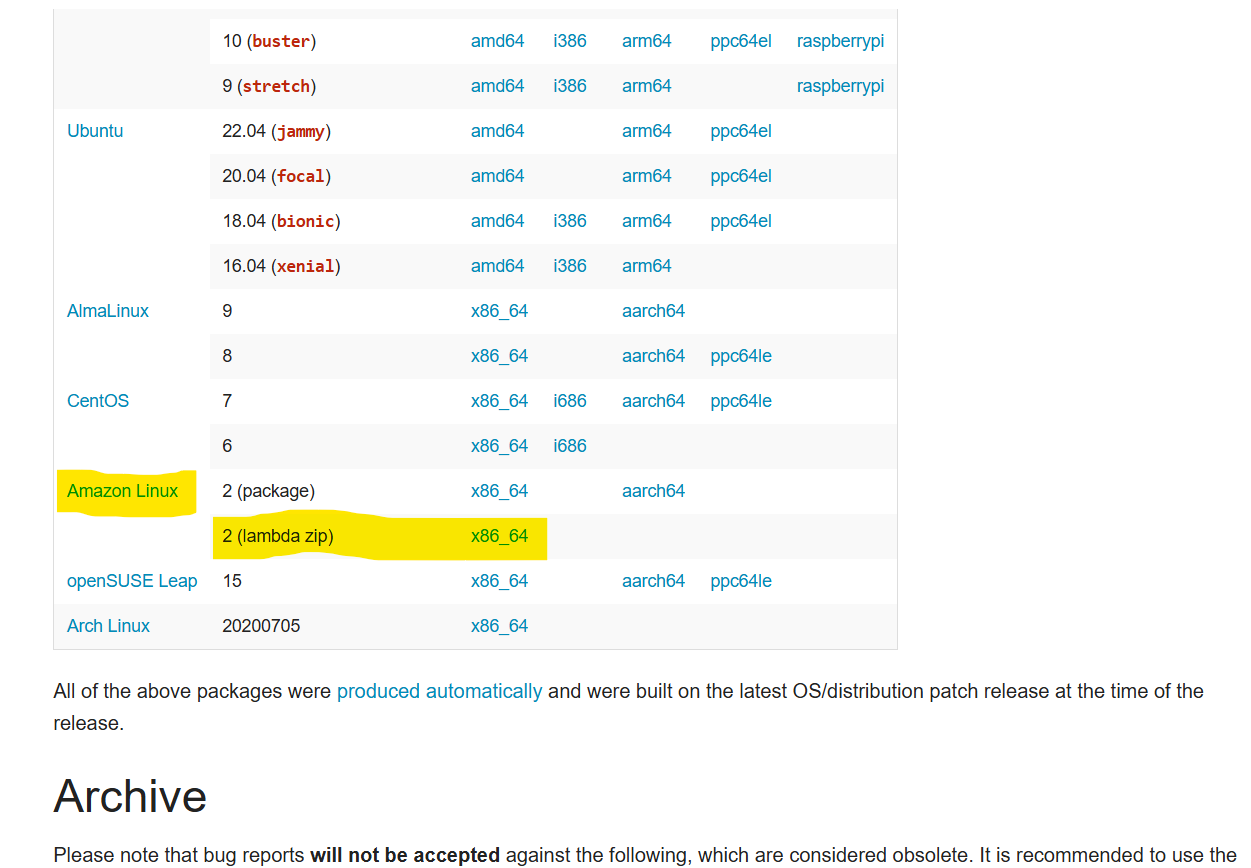
Download the wkhtmltopdf Lambda compatible zip file from official documentation and upload it as a layer.

Link: <https://wkhtmltopdf.org/downloads.html>

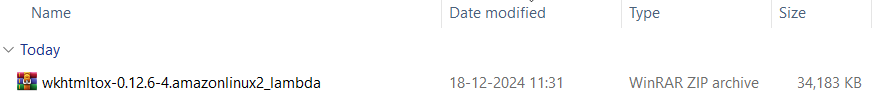
Or search for wkhtmltopdf downloads and reach the site.



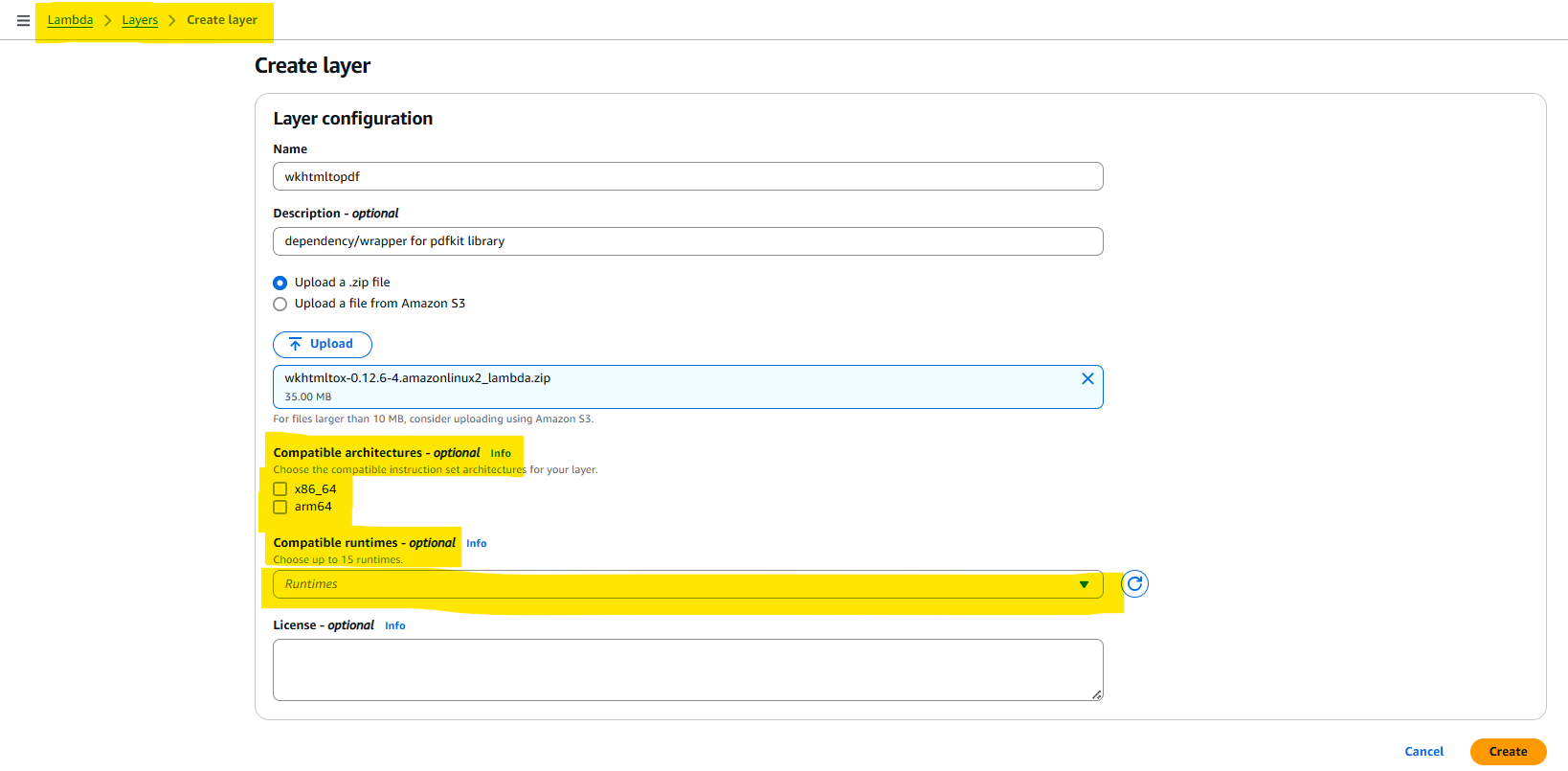
Choose the “Amazon Linux 2 (lambda zip) x86\_64” and download that zip file



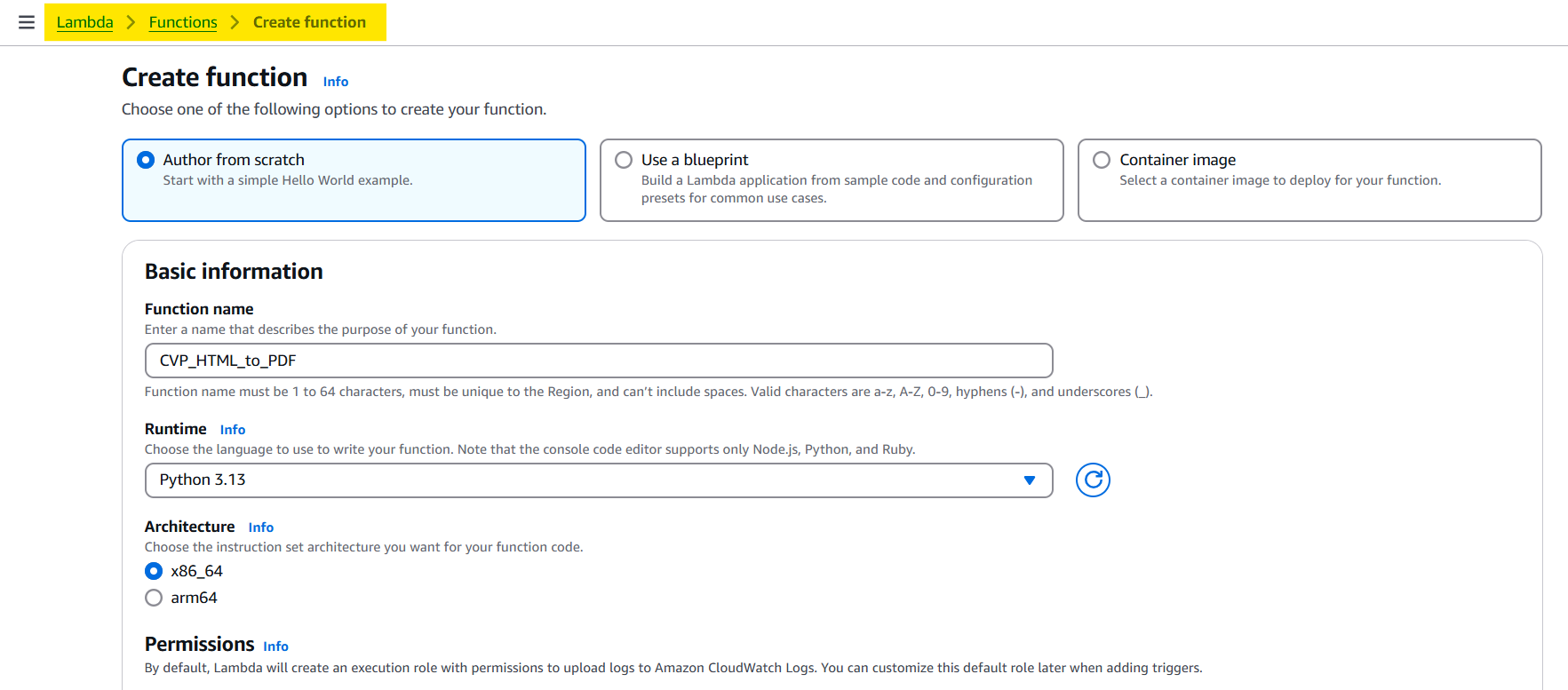
The downloaded file looks like the below which we will upload as a layer in AWS Lambda:



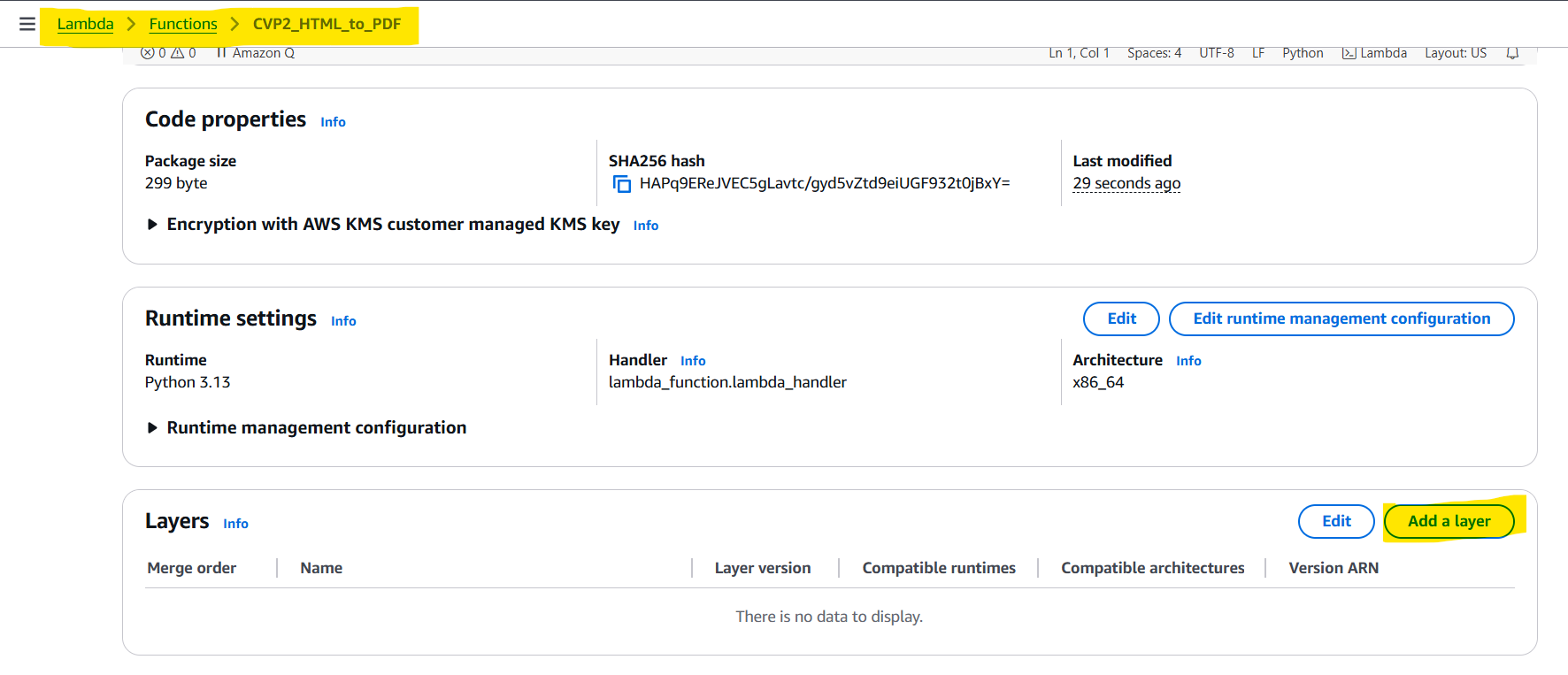
Create a Lambda Layer as shown below, Also don’t forget to keep “compatible architectures” and “compatible runtimes” blank

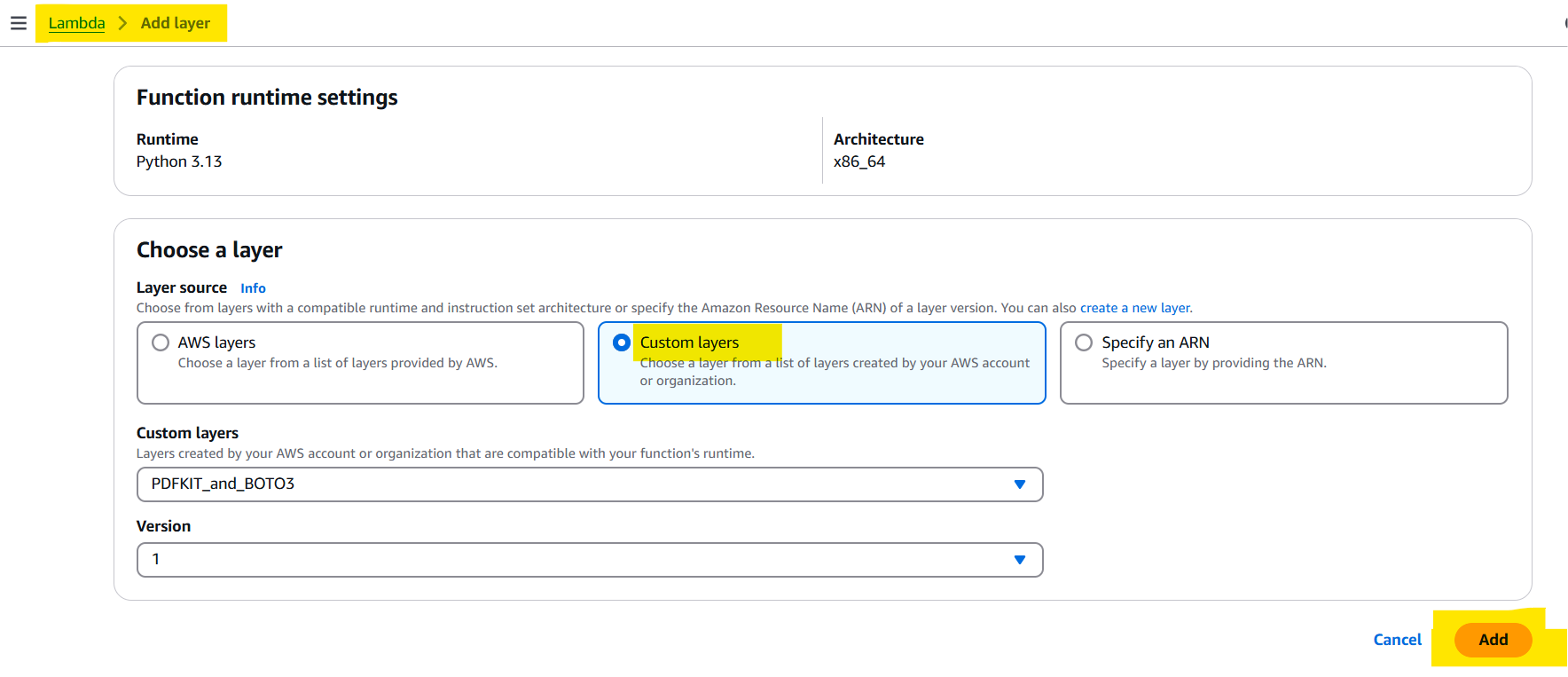


1. Now create a Lambda Function

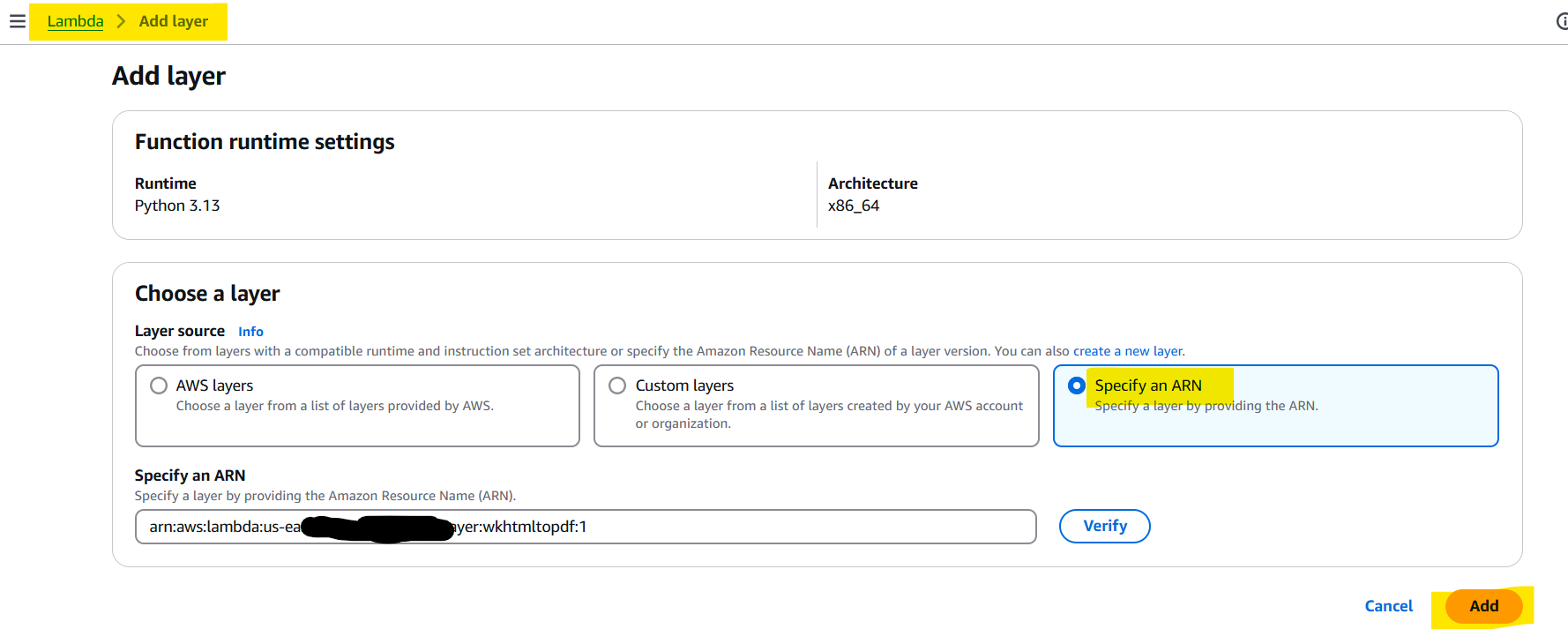


1. Don’t forget to attach an IAM policy to this Lambda function which has S3 read and write access.
2. Now add the created layers to this lambda.

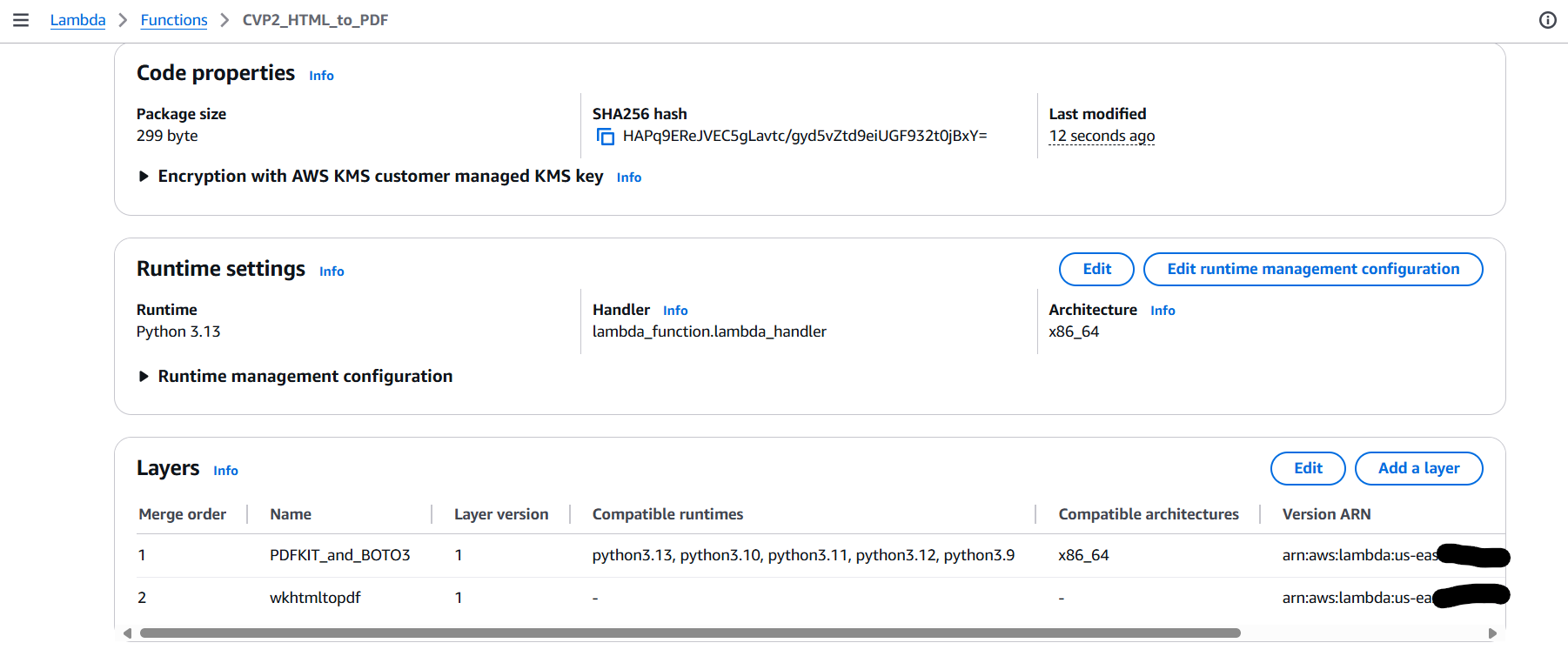




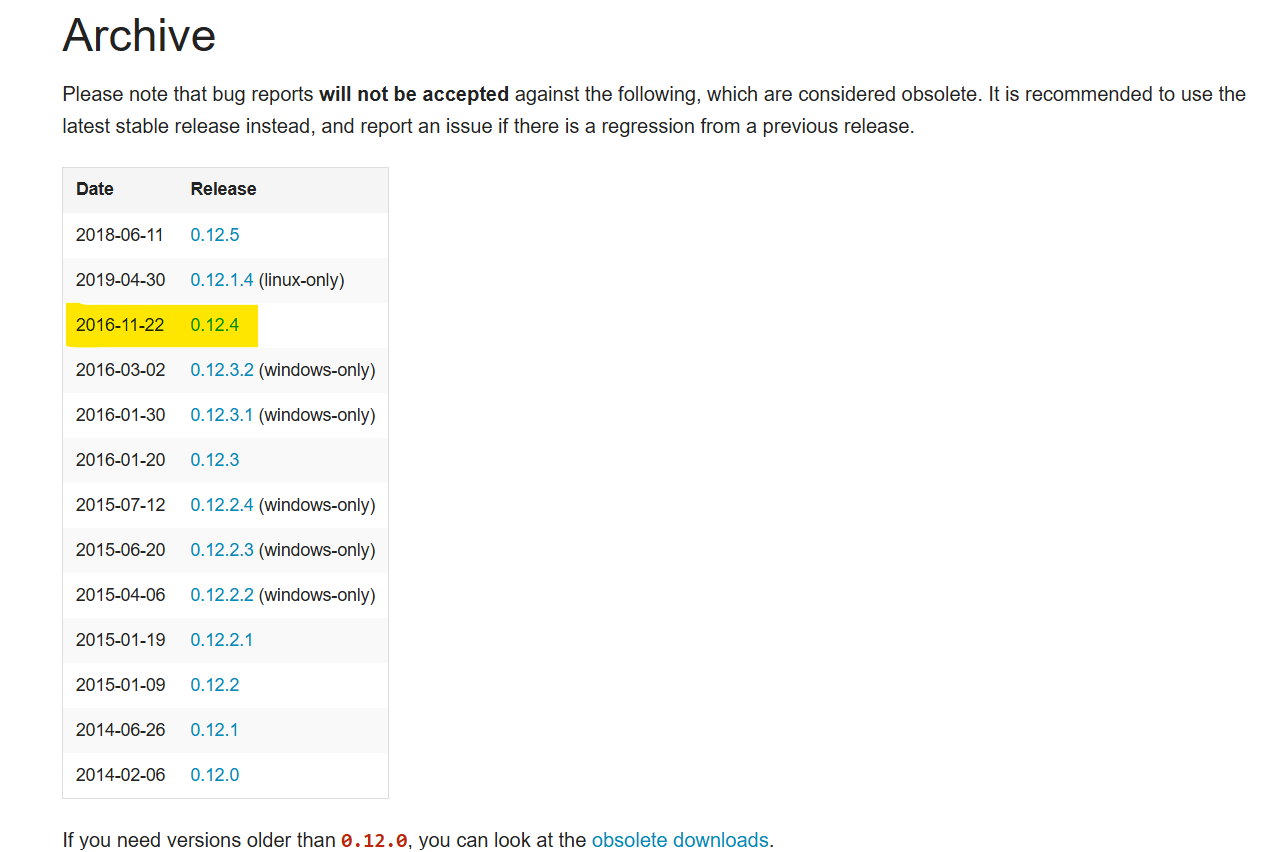
1. If the second layer is not visible to attach, then go to layer section and copy the ARN of the layer and use that to add layers in lambda function as below.



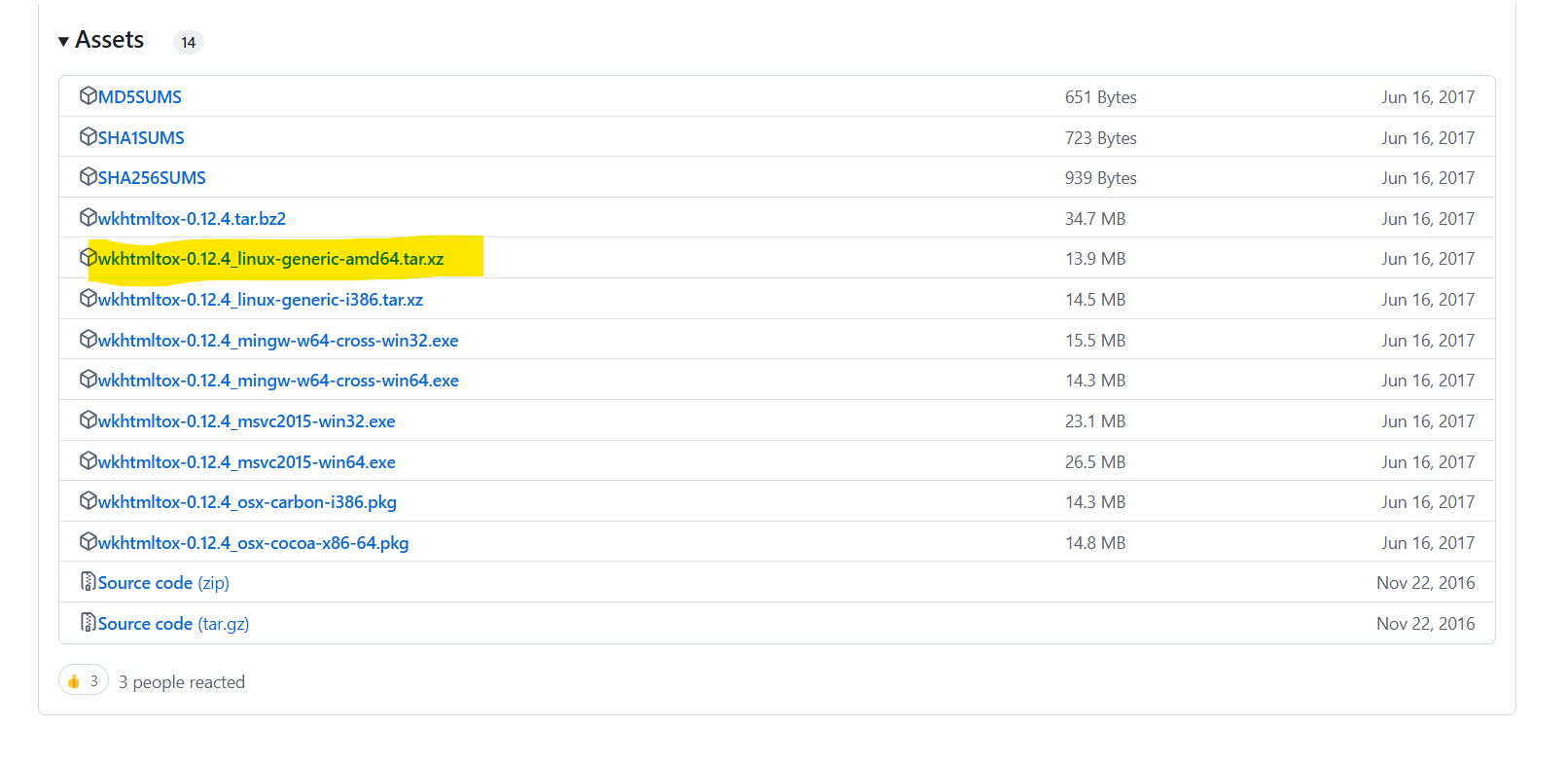
1. Now, both layers has been added



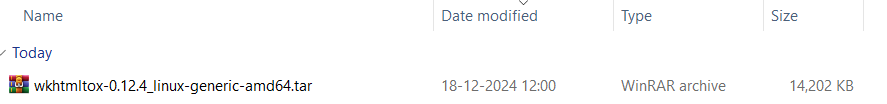
1. Now we need to install a wkhtmltopdf binary version Amazon Linux v0.12.4  
   Add the lambda\_function.py in this binary zip file and upload it in the AWS Lambda.
2. Go to the link: <https://wkhtmltopdf.org/downloads.html>  
   and download the binary file from here



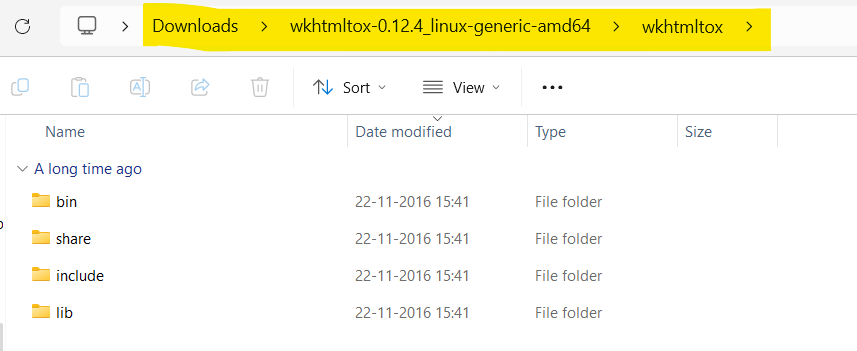
Download the below file:



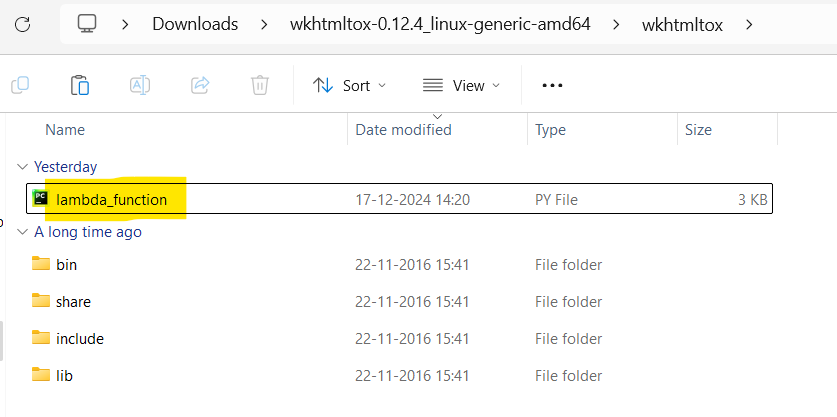
The downloaded file looks like below:



Unzip this tar file which looks like the below:



1. Now we are going to create a lambda\_function.py file here as shown below



1. The “lamda\_function.py” consists the below code:

1. import json

2. import boto3

3. import pdfkit

4. import os

5.

6. # Initialize the S3 client

7. s3\_client = boto3.client('s3')

8.

9.

10. def lambda\_handler(event, context):

11. # Source S3 bucket and key for the input HTML

12. input\_bucket\_name = 'cvp-2-bucket' # Replace with your input bucket name

13. input\_html\_key = 'input-html/input.html' # Replace with the path to the HTML file in the bucket

14.

15. # Destination S3 bucket and key for the generated PDF

16. output\_bucket\_name = 'cvp-2-output' # Replace with your output bucket name

17. output\_pdf\_key = 'output-pdf/output.pdf' # Path in the bucket where the PDF will be stored

18.

19. # Path to the wkhtmltopdf binary

20. wkhtmltopdf\_path = 'bin/wkhtmltopdf' # Adjust this path as needed (use Lambda Layer for wkhtmltopdf)

21.

22. try:

23. # Fetch the HTML file from the S3 bucket

24. response = s3\_client.get\_object(Bucket=input\_bucket\_name, Key=input\_html\_key)

25. html\_content = response['Body'].read().decode('utf-8') # Decode the content to string

26.

27. except Exception as e:

28. return {

29. 'statusCode': 500,

30. 'body': json.dumps(f"Error reading HTML file from S3: {str(e)}")

31. }

32.

33. try:

34. # Specify the wkhtmltopdf executable in pdfkit configuration

35. config = pdfkit.configuration(wkhtmltopdf=wkhtmltopdf\_path)

36.

37. options = {

38. 'orientation': 'Landscape'

39. }

40.

41. # Generate the PDF from the HTML content

42. pdf\_output = pdfkit.from\_string(html\_content, False, configuration=config, options=options)

43.

44. # Upload the generated PDF to the S3 bucket

45. s3\_client.put\_object(

46. Bucket=output\_bucket\_name,

47. Key=output\_pdf\_key,

48. Body=pdf\_output,

49. ContentType='application/pdf'

50. )

51.

52. return {

53. 'statusCode': 200,

54. 'body': json.dumps(f"PDF generated and uploaded to S3 at {output\_pdf\_key}")

55. }

56.

57. except Exception as e:

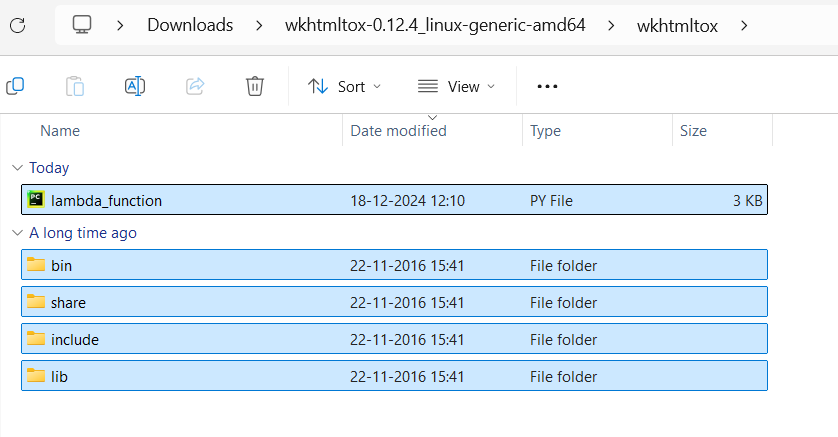
58. return {

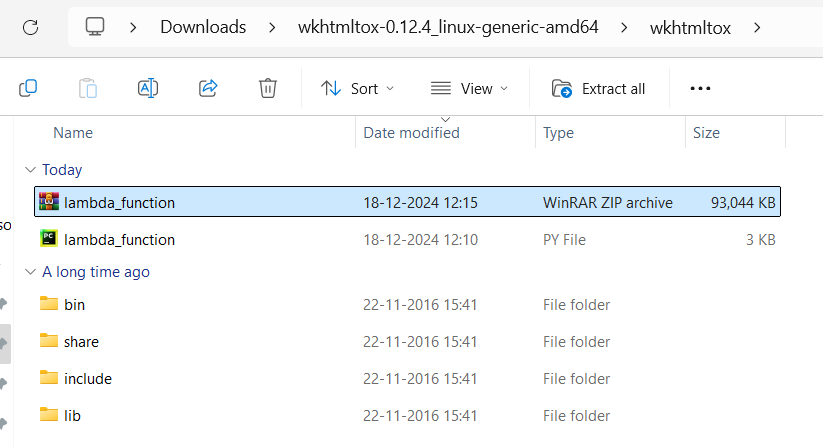
59. 'statusCode': 500,

60. 'body': json.dumps(f"Error generating PDF: {str(e)}")

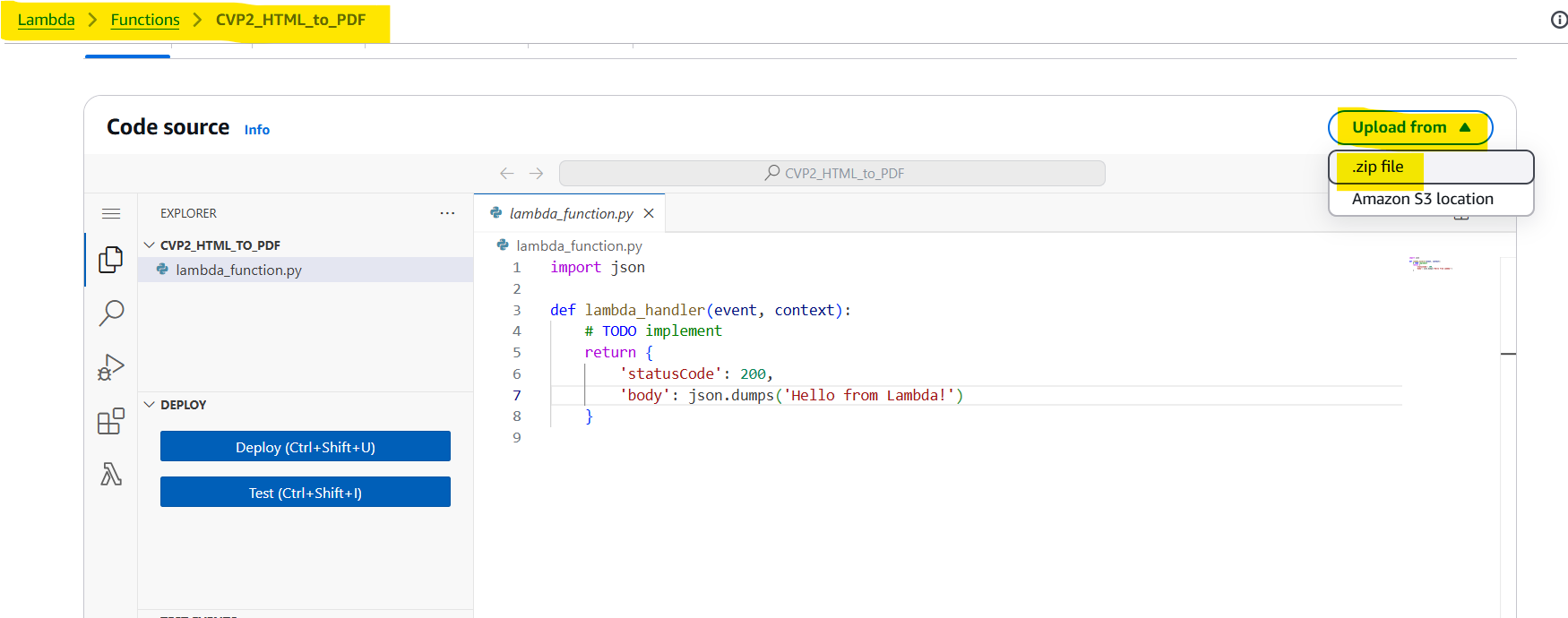
61. }

1. Now zip the below mentioned files:

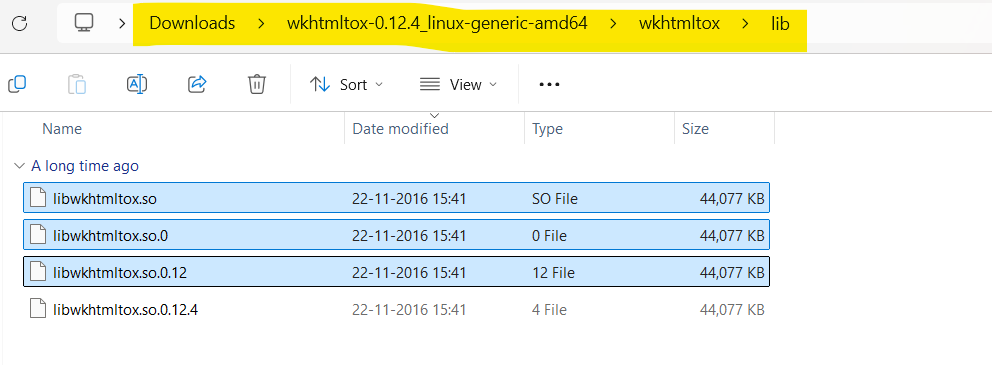


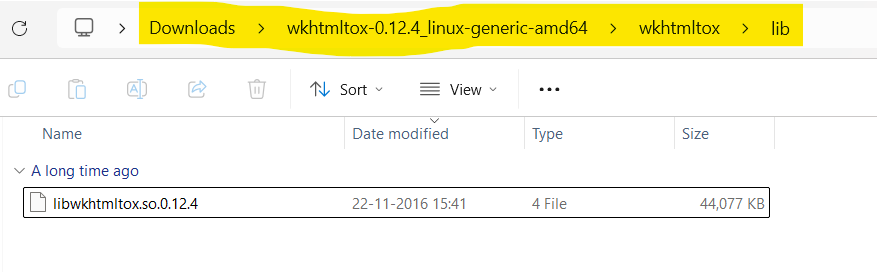


1. Now upload this to AWS Lambda

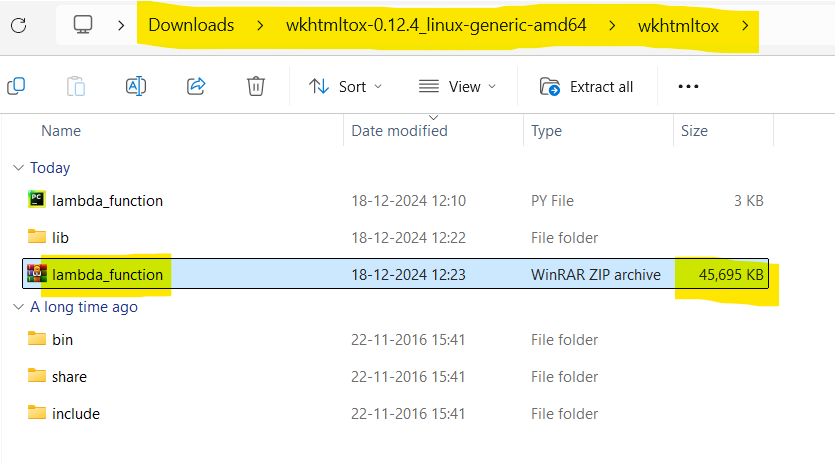


If you are facing issue like “file is too large to upload”, then you can delete the below mentioned files and again zip the folder.

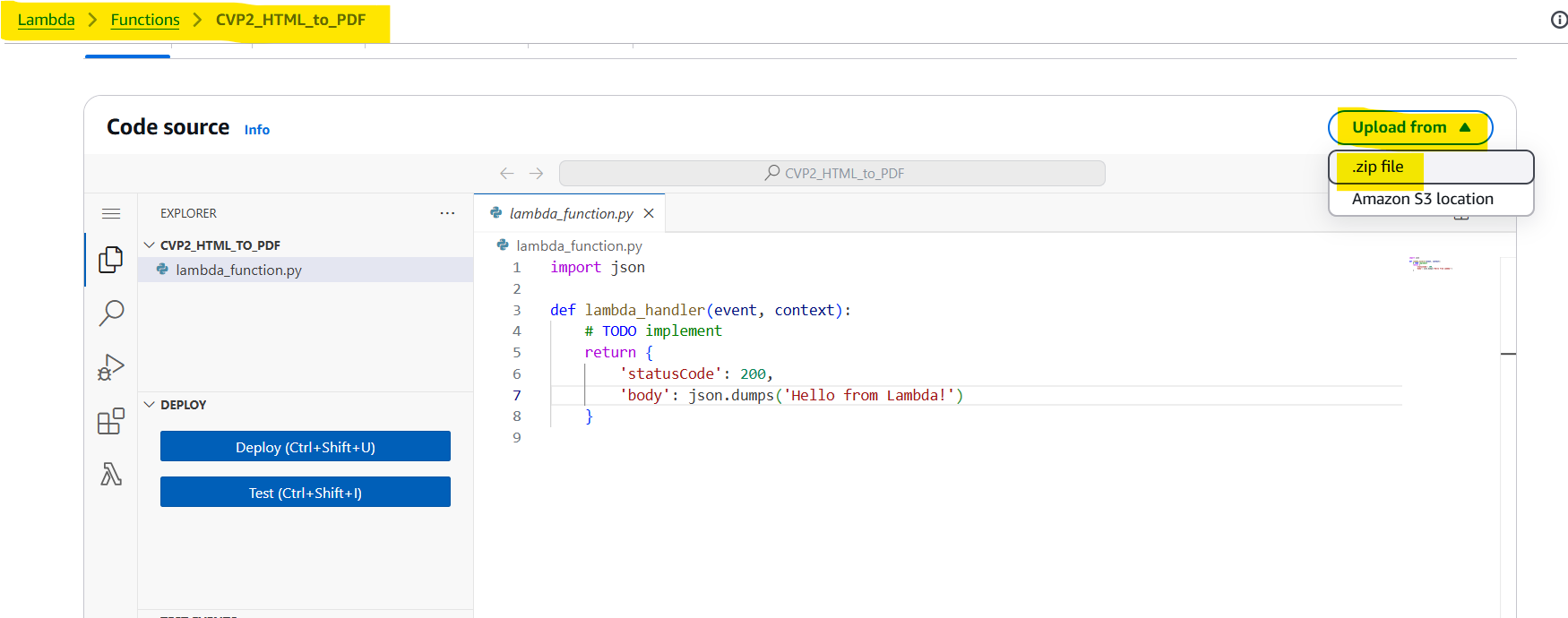




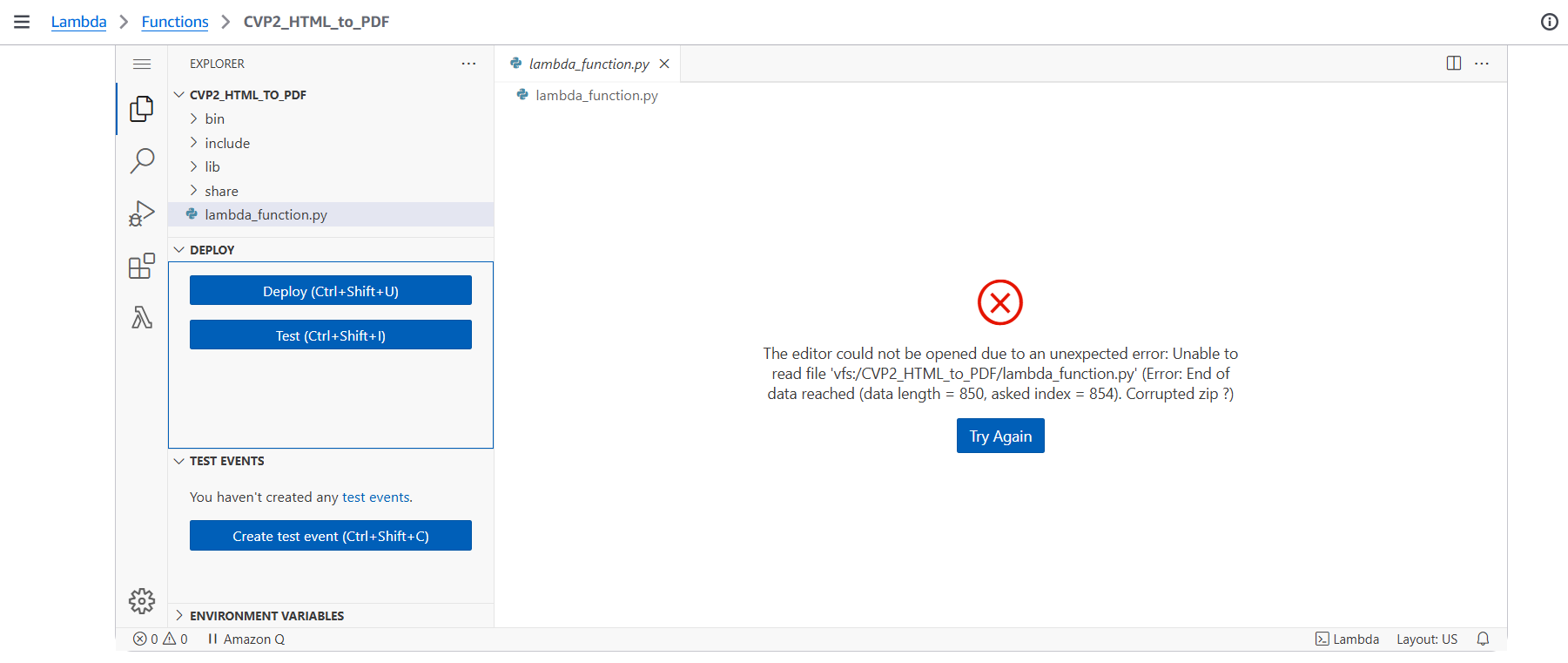
1. This will reduce the size of the zip file which you can upload easily



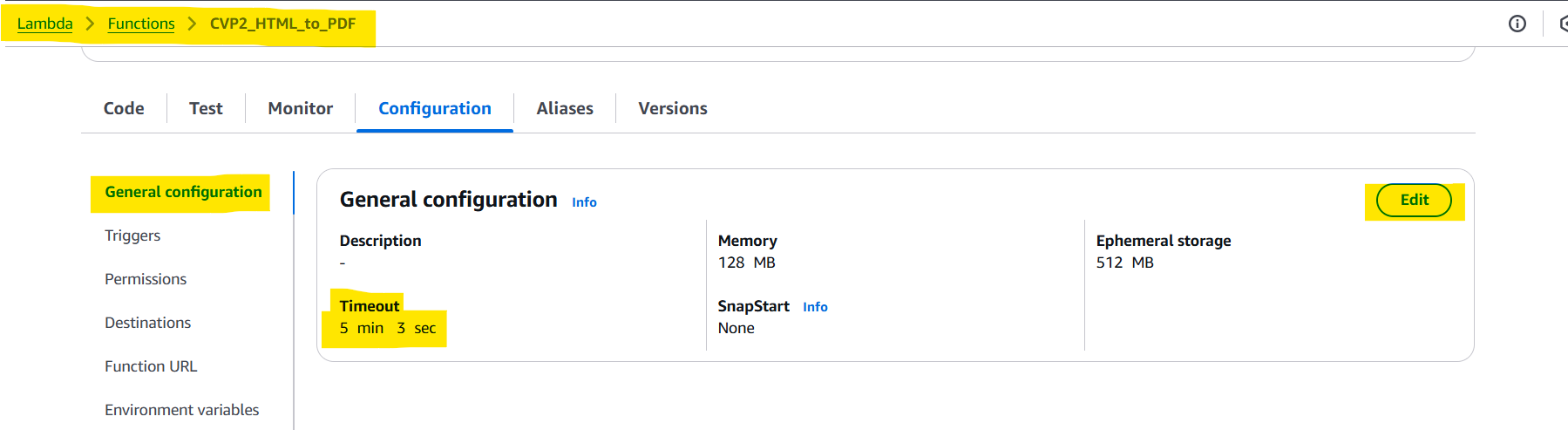
1. Now upload this zip to AWS Lambda



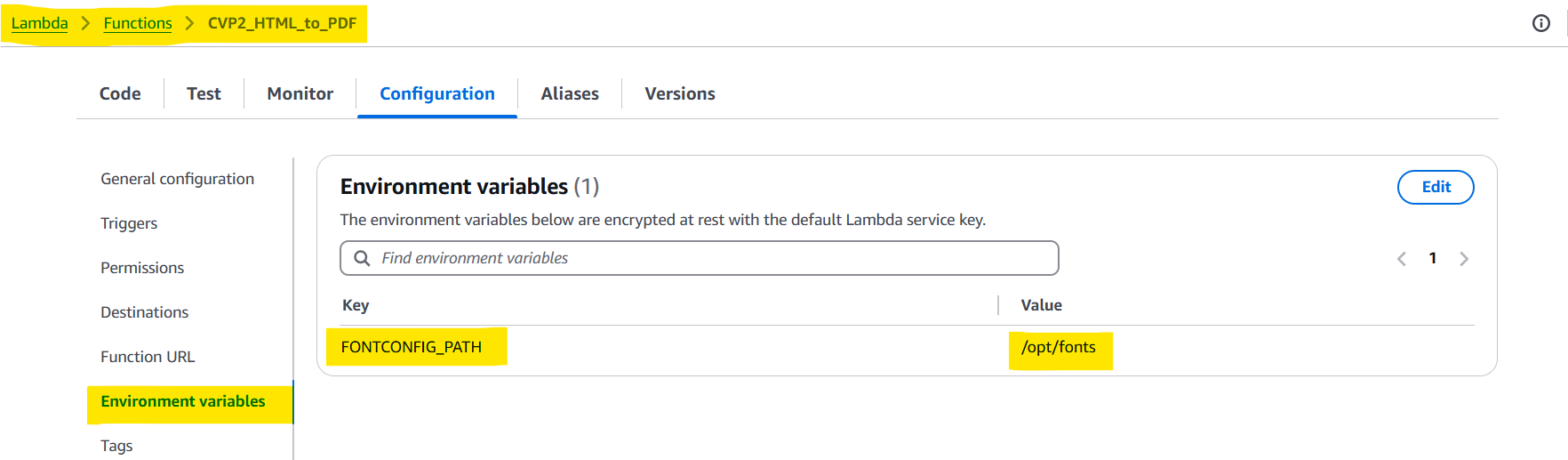
1. After uploading the zip file, it looks like below:  
   You will not be able to edit the “lambda\_function.py” file in the Lambda function if the size of the zip file is larger, it might show an error while viewing the file, but don’t worry we can still execute and run the lambda function.



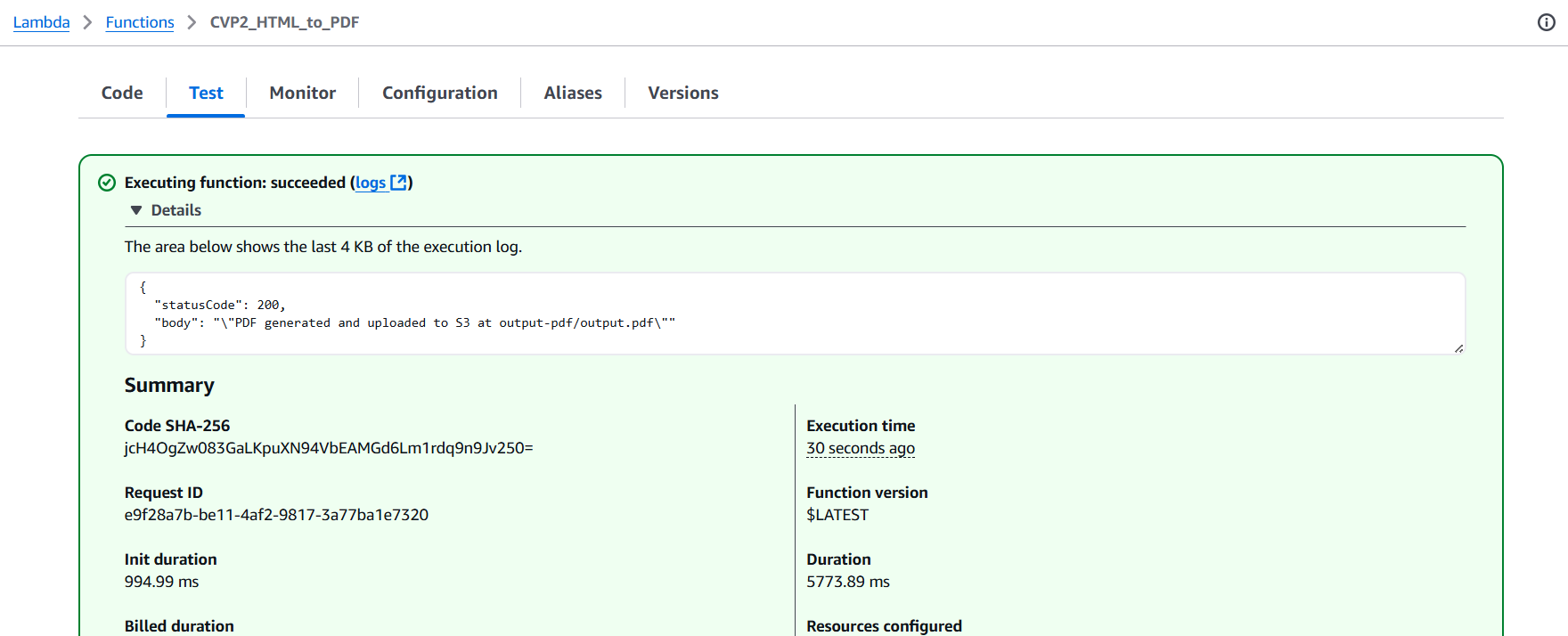
1. Go to AWS lambda and edit the timeout as shown below:



1. Also we need to add one environment variable path in Lambda  
   FONTCONFIG\_PATH : /opt/fonts



1. Now you can Test the Lambda function by clicking on the “Test” button and the function will be executed.



1. You can go and check the output S3 bucket, the PDF file will be generated

