Problem Statement: Create an AWS Lambda Function that takes a message from an event and stores it in the S3 Bucket as a text file

Project Overview:

1. A message is triggered – manually for now
2. AWS lambda function will process the message
3. The python function will store the message as a text file in the S3 Bucket

Pre-requisites:

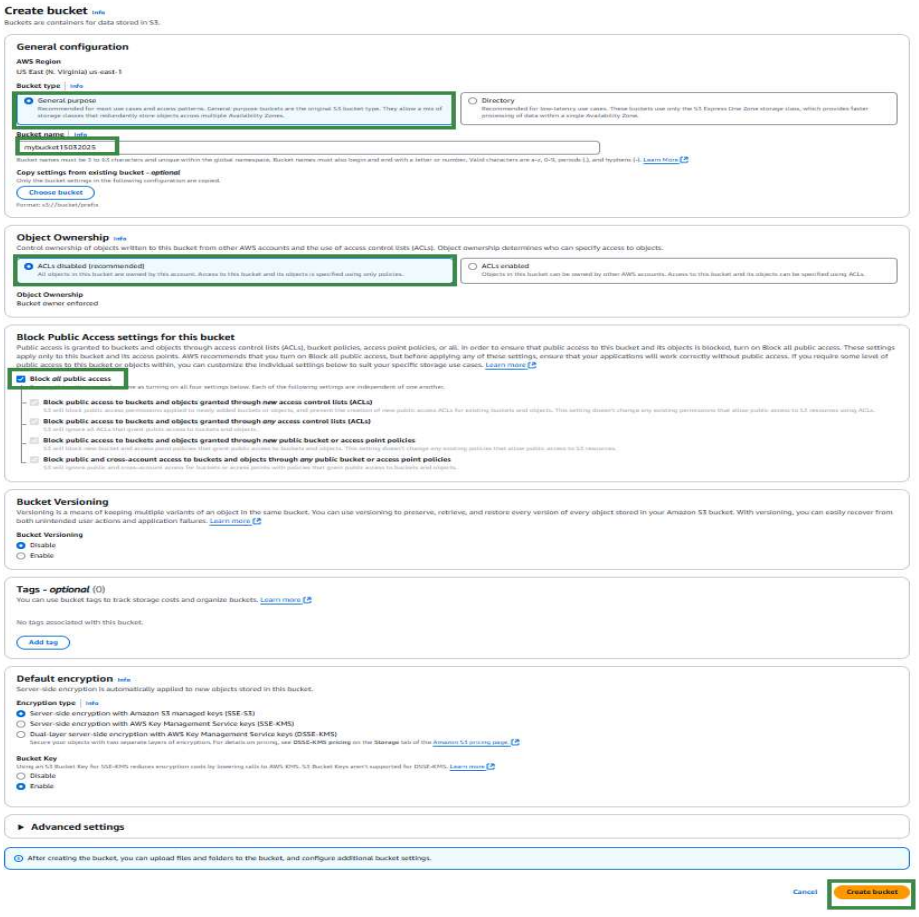
1. Amazon Free Tier Account.
2. Knowledge on how to create IAM users and add permissions.

Step-By-Step Instructions:

Step 1:

1. Login into AWS console
2. Navigate to S3 bucket
3. Click on create bucket

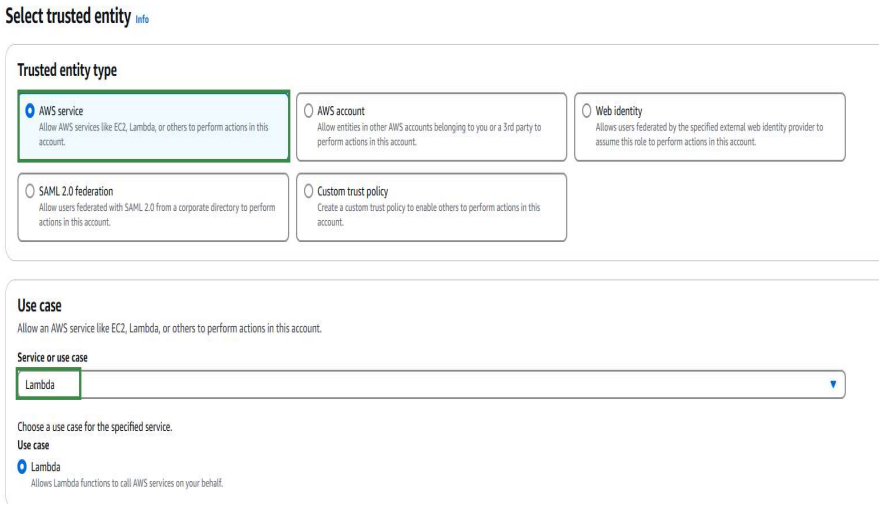
4. Select General Purpose
5. Provide a unique bucket name
6. Leave the rest as default settings and click on create



7. The bucket is created successfully

Step 2:

8. Navigate to IAM and click on create a role

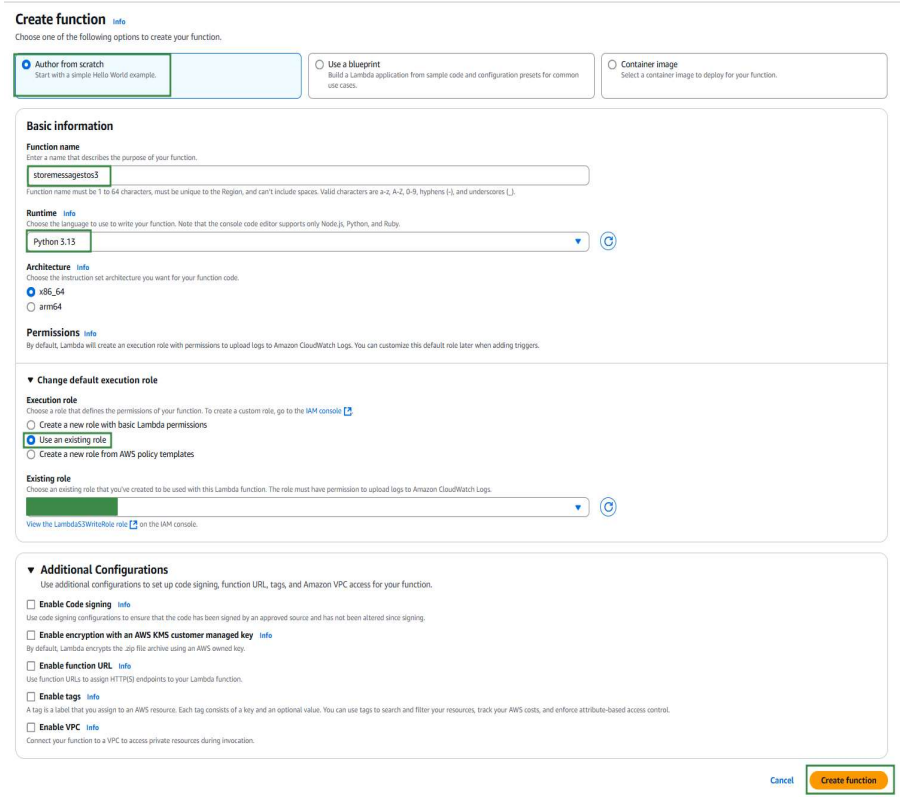


9. Locate the permission **AmazonS3FullAccess**

10. Click on Next and provide a role name
11. Create a role

Step 3:

12. Now we will create a lambda function using python
13. Click on create function
14. Select **Author from scratch**
15. Provide a name to the function
16. Select runtime as python from the dropdown
17. Now attach the IAM role created in previous step
18. Expand change default execution role
19. Select use existing role
20. From the drop down select the IAM role
21. Click on create function



Step 4:

22. In the function code editor past the code.

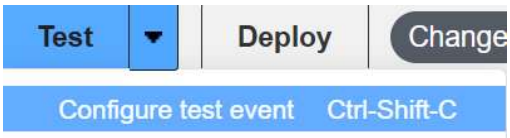
s3upload.py

23. The code will be available in the GitHub for download.

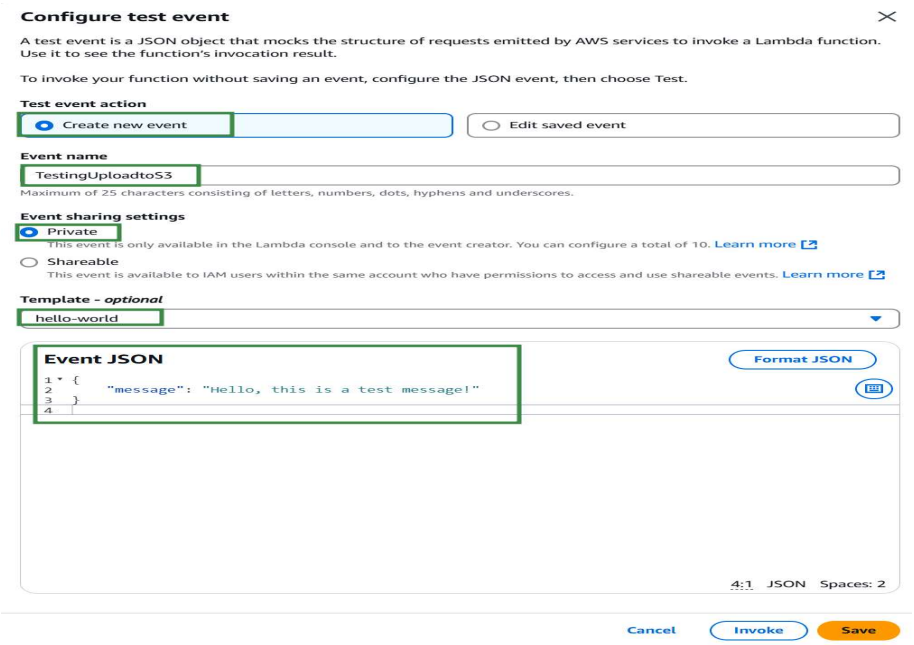
```
lambda_function   Environment Var

import boto3  # AWS SDK for Python (used to interact with AWS services)
import datetime  # Module to handle timestamps for unique file names

# Initialize the S3 client
s3 = boto3.client('s3')

# Define the name of the S3 bucket where messages will be stored
BUCKET_NAME = "[redacted]"  # Replace with your actual bucket name

def lambda_handler(event, context):
    """
    AWS Lambda function to store a message in an S3 bucket as a text file.
    This function gets triggered manually or via an event (future use case).

    Parameters:
    event (dict): Event data passed to the Lambda function (not used in this basic version).
    context (object): Metadata about the function execution (not used here).

    Returns:
    dict: Response containing HTTP status code and confirmation message.
    """

    # Message to be stored in S3 (Static for now; can be made dynamic later)
    message = "Hello, this is a test message!"

    # Generate a unique timestamp for the file name to avoid overwriting existing files
    timestamp = datetime.datetime.now().strftime("%Y-%m-%d_%H-%M-%S")

    # Define the file name where the message will be saved inside the "messages/" folder
    file_name = f"messages/message_{timestamp}.txt"

    # Upload the message to the specified S3 bucket
    s3.put_object(
        Bucket=BUCKET_NAME,  # The target S3 bucket
        Key=file_name,       # The path (folder + file name)
        Body=message         # The actual message content to be stored in the file
    )

    # Return a success response
    return {
        'statusCode': 200,
        'body': f"Message stored in {file_name}"  # Confirmation message
    }
```

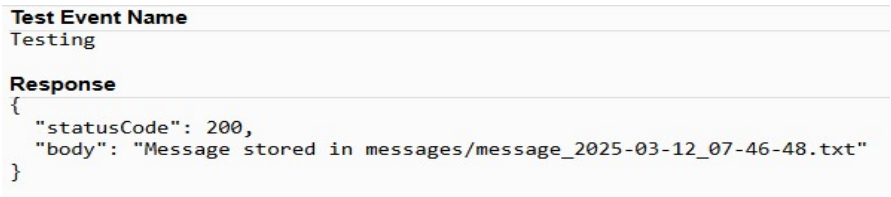
Step 5:

24. Within the function, test the code and check if the messages are stored in the S3 bucket
25. **Click on deploy**
26. Click on Test and configure test event



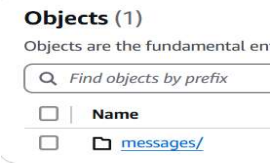
27. Click on Create New Event
28. Provide an Event Name
29. In the template select Hello World
30. In the Event JSON, replace the code
31. Click on Invoke
32. In the execution results the response will be 200 Status Code





Step 6:

33. Navigate to S3 Bucket
34. You will find a object called messages



35. Click on messages and text file is uploaded

36. Download the message and the content is available as mentioned above. **This is a manual process**, and it can be **done dynamic using the event.get method**.

Step 7:

37. Replace the single line code in the function as shown

```
# Message to be stored in S3 (Static for now; can be made dynamic later)
#message = "Hello, this is a test message!"
message = event.get("message", "Default Message")
```

38. Deploy the code again

39. Create a new test event and in the event json give any message and click on invoke. It will be stored as text files in S3 as shown

```
Hello, this is a test message123456789!
```