Project Title: Professional Text Enhancer using Amazon Bedrock and Claude

Overview:

This project is a serverless AI powered text rewriting tool that takes the user content and returns a professional version using Claude model through Amazon bedrock. The solution can be integrated into any productivity or enterprise writing assistant.

Architecture:

1. Postman – Send a POST request with raw text
2. AWS Lambda – Python code – event driven - API gateway
3. AWS Claude model to rewrite the input text
4. S3 – Stores input and output logs in JSON format for auditing
5. IAM – Grant permissions to invoke Bedrock and write to S3

Result:

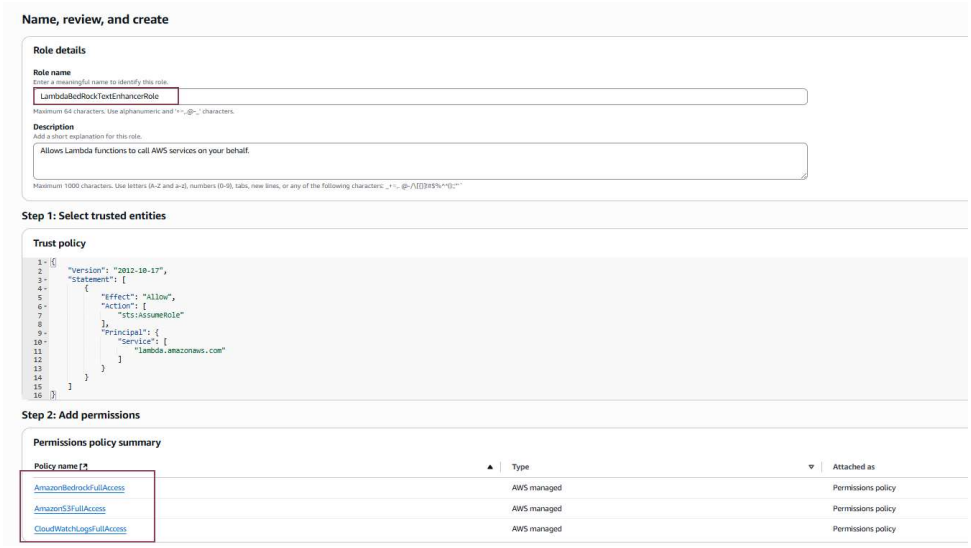
1. Accepts Plain English sentences
2. Sends to Claude to re-write it professionally
3. Returns the response handling errors gracefully and missing input
4. Original content and re-written content with timestamp to S3

Additional Notes

1. Use region us-east-1
2. Understand the Quotas Amazon Bedrock

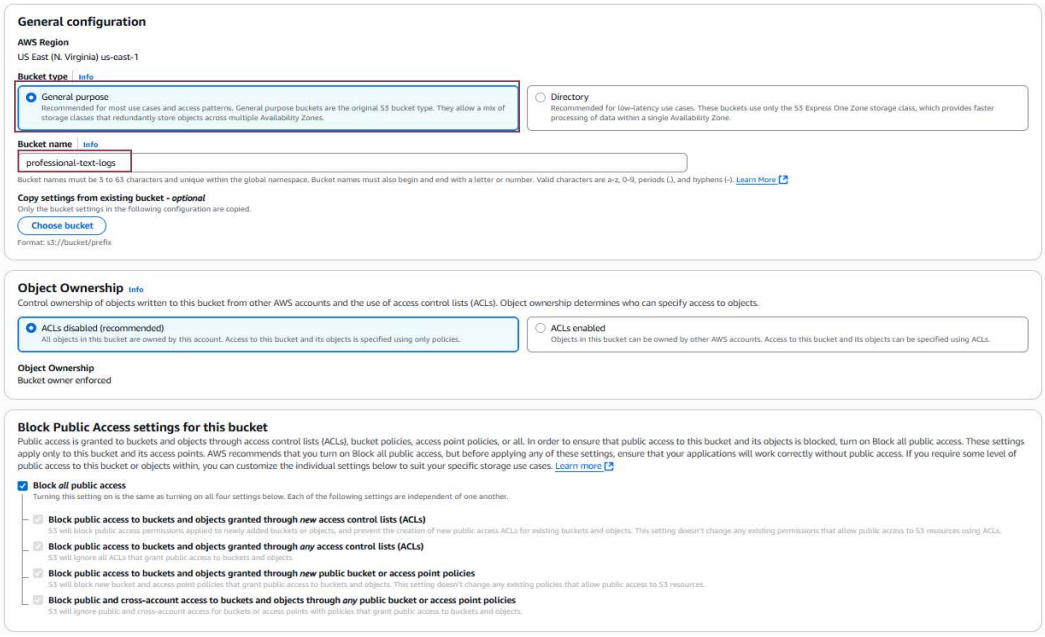
Step 1

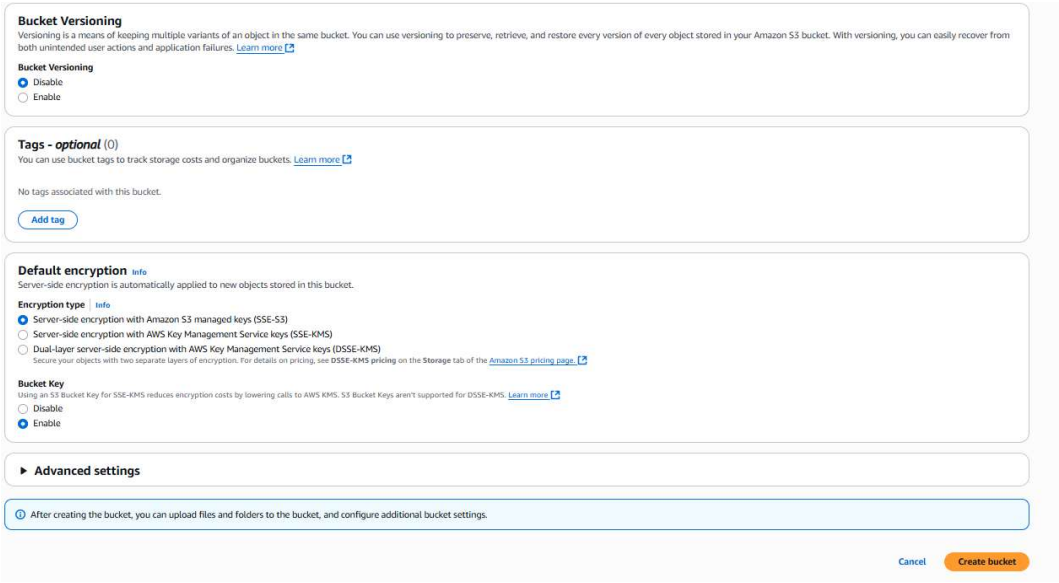
1. Go to IAM Roles
2. Create a Role
3. Add the Following Permissions
    a. AmazonBedRockFullAccess
    b. AmazonS3FullAccess
    c. CloudWatchLogsFullAccess
4. Provide a Name – LambdaBedRockTextEnhancer
5. Click on Create Role



Step 2

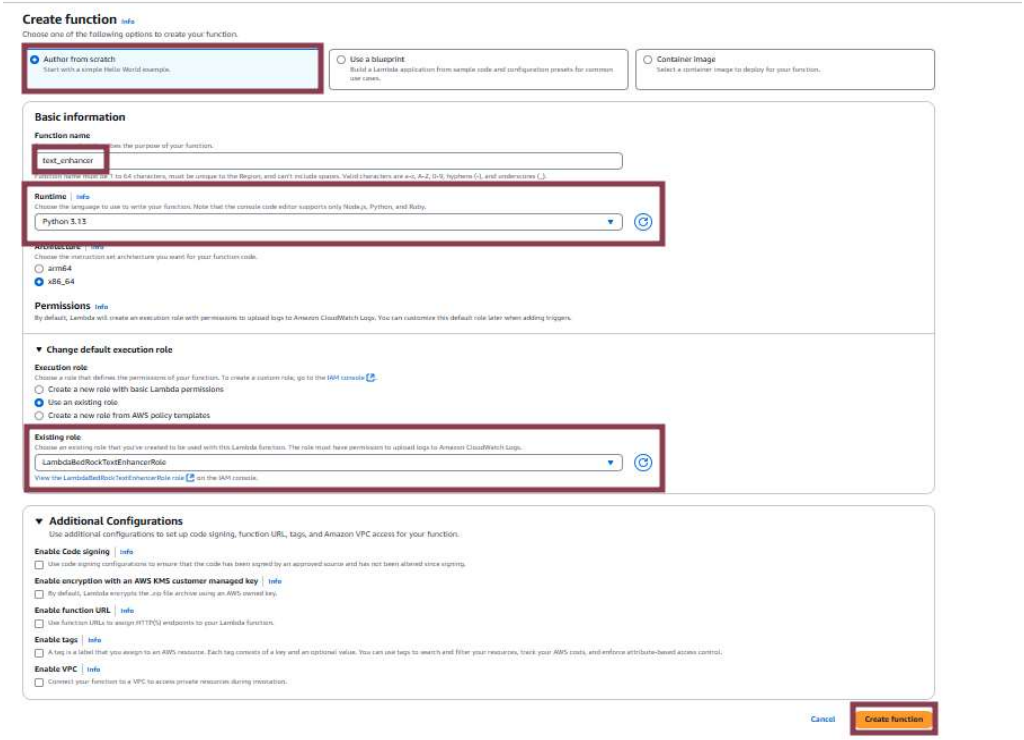
1. Create a bucket
2. Select General Purpose
3. Provide a name
4. Click on Create Bucket

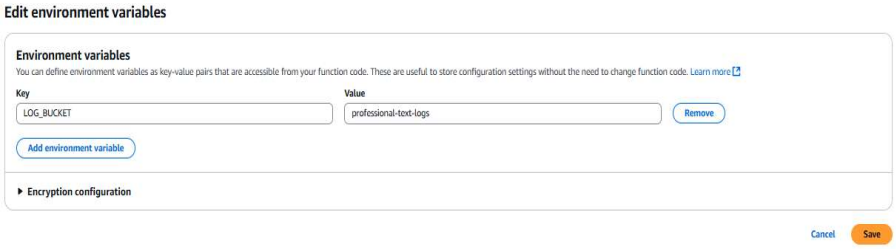




Step 3

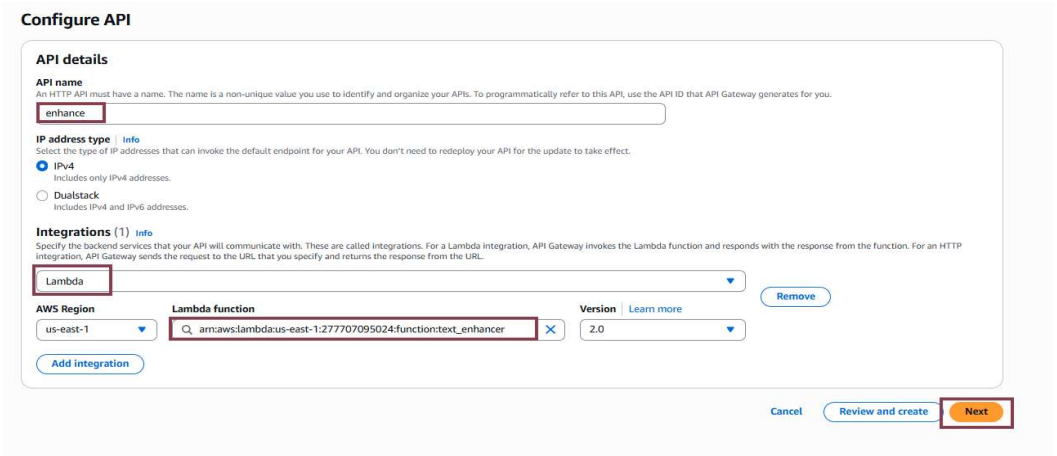
1. Create a lambda function
2. Select Author From Scratch
3. Provide a Function Name – Call it text-enhancer
4. Select runtime as Python
5. Select the IAM role
6. Click on Create Function
7. Click on the Function
8. Navigate t Configuration
9. Click on Environment Variables
10. Add the Key LOG_BUCKET
11. Add the value S3 Bucket Name
12. The lambda function will automatically resolve to the S3 bucket time during runtime
13. Deploy the code

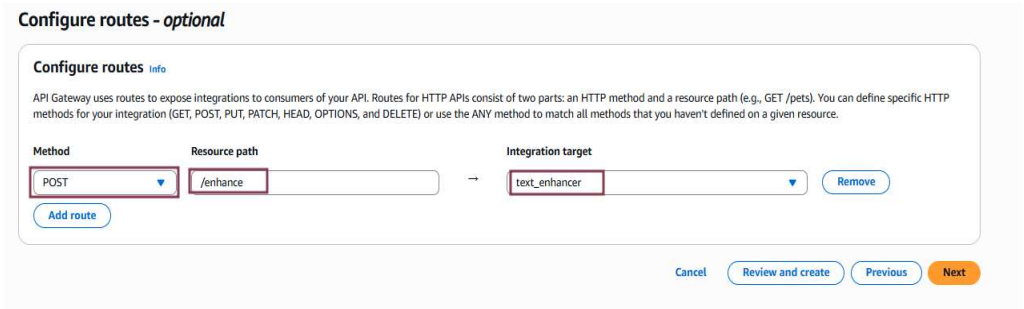


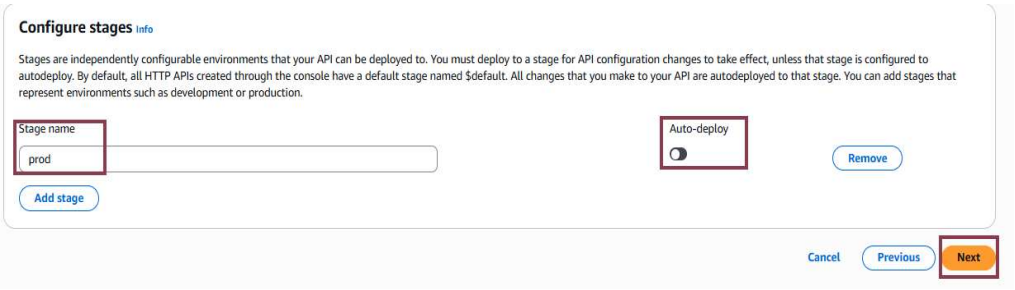


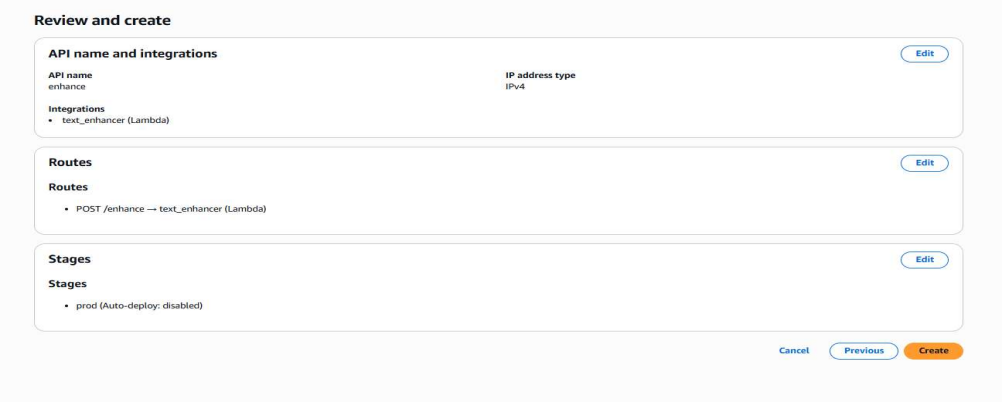
Step 4

1. Go to API Gateway
2. Click on HTTP API
3. Click on Create
4. Set Route as Post
5. Ensure the API name is consistent with the project to avoid confusion will creating routes or configuration
6. Click on Add Integrations
7. Select the region
8. Select the Lambda Function
9. Click on Next
10. Select Method as POST
11. Click on Integration Target – Select from Drop Down – The lambda function name
12. Provide the resource path as /enhance
13. Click on Next
14. Deploy the API




Configure routes - optional
Configure routes Info
API Gateway uses routes to expose integrations to consumers of your API. Routes for HTTP APIs consist of two parts: an HTTP method and a resource path (e.g., GET /pets). You can define specific HTTP methods for your integration (GET, POST, PUT, PATCH, HEAD, OPTIONS, and DELETE) or use the ANY method to match all methods that you haven't defined on a given resource.
Method
Resource path
Integration target
POST
/enhance
text_enhancer
Remove
Add route
Cancel
Review and create
Previous
Next

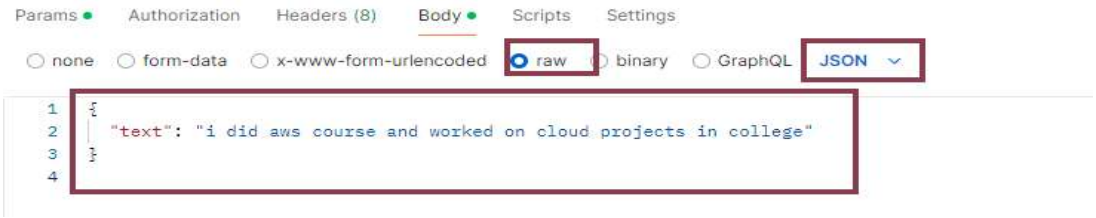

Configure stages Info
Stages are independently configurable environments that your API can be deployed to. You must deploy to a stage for API configuration changes to take effect, unless that stage is configured to autodeploy. By default, all HTTP APIs created through the console have a default stage named $default. All changes that you make to your API are autodeployed to that stage. You can add stages that represent environments such as development or production.
Stage name
prod
Auto-deploy
Remove
Add stage
Cancel
Previous
Next


Review and create
API name and integrations
Edit
API name
enhance
IP address type
IPv4
Integrations
text_enhancer (Lambda)
Routes
Edit
Routes
POST /enhance → text_enhancer (Lambda)
Stages
Edit
Stages
prod (Auto-deploy: disabled)
Cancel
Previous
Create

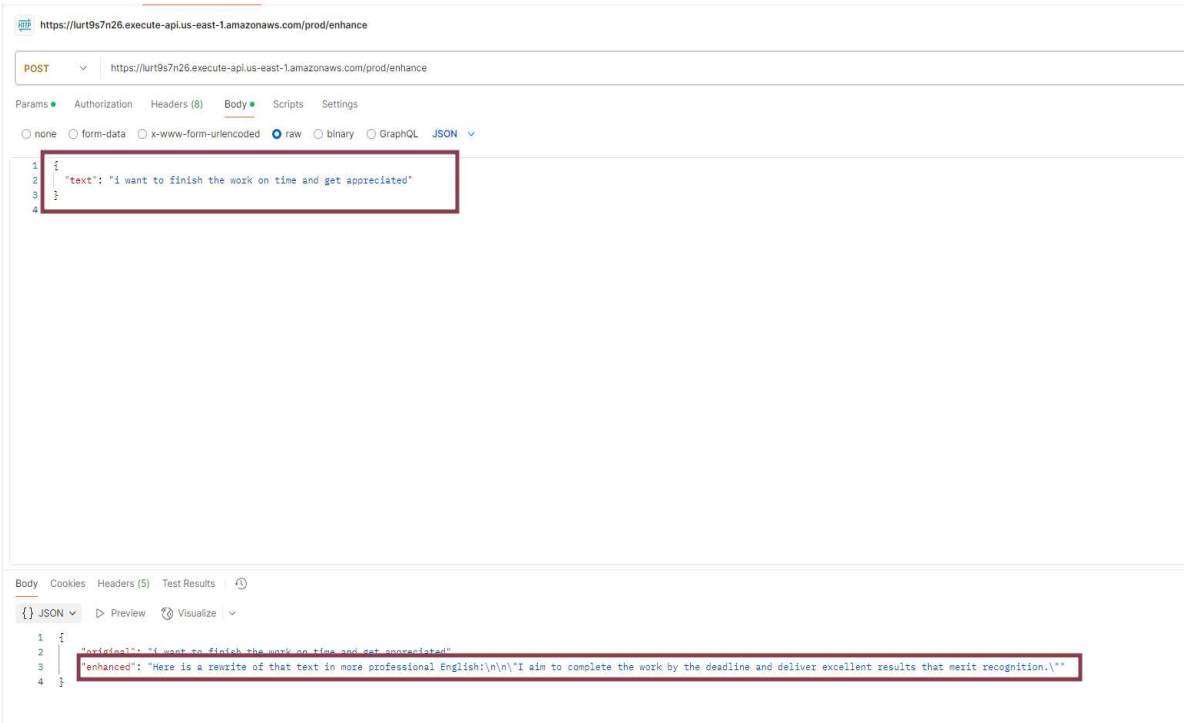
Step 5

1. Copy the Invoke URL from the Deploy > Stage
2. Open postman
3. This is POC and will test using POSTMAN
4. Go to the section Body
5. Select Raw
6. Select JSON
7. Enter the content in the body as shown
8. You should get a response
9. If you get an internal server error, then ensure the Amazon Bedrock model is enabled
10. Note down the model ID and that ID will be used in the lambda function code

Sample JSON OUTPUT



Enhanced Text from Amazon Bedrock



S3 – Log

text-enhance.py

C: > Users > svrar > OneDrive > 1.Study_Plan > 6.Intellipat > Lab_Assignment > B.Other_Labs > Own_Projects > Project_10 > text-enhance.py

```python
import boto3
import json
import uuid
import datetime
import os

bedrock = boto3.client("bedrock-runtime", region_name="us-east-1")  # Use correct region
s3 = boto3.client("s3")
BUCKET = "professional-text-logs"  # Replace with your actual bucket name
[redacted]
            body=json.dumps(payload),
            modelId="anthropic.claude-v2:1",
            accept="application/json",
            contentType="application/json"
        )

        response_body = json.loads(response['body'].read())
        enhanced_text = response_body.get("completion", "").strip()

        log_entry = {
```