TrustCheck

A Chrome Extension for Evaluating Online Product Description Legitimacy

Sriram B. Swami – 22BCE5103

Meghna Varma – 22BCE5085

AO Navin Kumar – 22BCE1020

Aakash K – 22BCE5157

Under the mentorship of Dr. Priyaadharshini M,

School of Computer Science and Engineering (SCOPE),

VIT Chennai

| Objectives

The primary objective of our project is to assess the legitimacy of the product descriptions posted on e-commerce websites such as Amazon by comparing them with an overall aggregate of actual customer reviews in an aspect-based fashion. The system utilizes web scraping and aspect-based sentiment analysis to try to check whether the product descriptions accurately reflect the experiences and expectations of customers. In doing so, this project aims for:

- **Increase in Customer Satisfaction:** Help customers know which products would meet their expectations, saving them valuable time spent going through comments, allowing for informed purchasing decisions and increasing their satisfaction with their experience.
- **Decrease in False Advertisements:** Improve transparency among sellers by flagging products that failed to meet their description standards upon purchase.
- **Reduction in Product Returns:** Minimize misinformed purchases, hence reducing overall product returns that were due to dissatisfaction, benefitting the time of the customers as well as the sellers.
- **Seller-Buyer Trust Factor:** Generate a degree of trust among sellers and buyers with improved transparency, pushing towards a more reliable and pleasing marketplace.

Benefitting customers as well as legitimate sellers, we believe our solution pushes for a more credible marketplace and allows for a happier experience for those shopping on e-commerce websites.

| List of Modules

1. Web scraping module
2. API Gateway Module
3. Data Orchestrator Module
4. Preprocessing module
5. NLP Module

| Module Description

1. **Web scraping module**

This contains the browser extension which makes up the frontend ui. The browser extension takes an optional “aspect” (eg: if we’re considering a smartphone, viable aspects could be battery-life, camera quality, durability etc.) as its parameter which it bases the sentiment analysis off of. If no aspect is provided, it sentiment analyses all available reviews.

It then scrapes the reviews, formats them into json format and then sends them to the lambda function using AWS API Gateway.

This is hosted on an AWS EC2 Server to ensure resources are allocated efficiently, expanding during peak times and contracting during periods of low activity, thereby optimizing costs while maintaining performance.

2. API Gateway Module

We use an HTTP gateway which connects directly to the lambda function hosting the endpoint.

API Gateway acts as a secure endpoint to transmit data to the lambda function. It supports **SSL/TLS policies** to secure communications between clients and the API Gateway. These policies enforce encrypted data transmission, ensuring the confidentiality of data exchanged.

API gateway also enforces the integrity of data, ensuring that the transmitted data consists of well-formed JSON payloads and contains the required fields.

3. Data Orchestrator module

This module is implemented using the lambda function which hosts the SageMaker model's endpoint. It acts as a backend processor bridging the gap between API gateway and SageMaker. It also handles on demand scaling, processing requests efficiently.

It routes the JSON input from the API gateway, sends the data to SageMaker and returns the processed results back to the browser extension. A timeout mechanism is also implemented, and the limit is set to 5 minutes to manage errors, such as model timeouts or invalid inputs. If an output is not received within this time period, the request is terminated, and the next query is executed.

An environment variable is used to map the endpoint name to a key, ensuring data abstraction and security. Policies and permissions are also configured to allow lambda to access the endpoint.

4. Preprocessing module

Training data stored in the s3 bucket is retrieved for preprocessing using the python library: boto3. Various preprocessing techniques are implemented to clean the data removing non-essential characters like emojis and special characters. Further reviews written in languages other than English, are discarded to remove dataset noise.

Then tokenization is performed next, splitting complex sentences into individual words or sub-words. Sequences of varying lengths are padded to standardise their lengths as the model requires uniform length sequences.

Stop-word removal is also applied, removing connector words which typically don't add much meaning to the sentence like "is", "the", "a" etc. This pipeline ensures that the model is trained on clean and consistent data.

5. NLP Module

When it comes to training, the data is split in an 80:20 ratio with 80% making up the train set and 20% making up the test set.

Model structure is created using a simpleRNN as the base model to capture sequential dependencies in the text. Further, embedding, flatten and dense layers are also added sequentially to convert words

to dense vectors, converting high dimensional data to lower dimensional data or dimensionality reduction and to convert data into the desired output format (Either -1,0 or 1 where -1 corresponds to negative, 0 to neutral and 1 to positive sentiment).

A label encoder is then used to map the obtained integer to its respective sentiment, printing along with it a confidence score, indicating the model's certainty of its answer.

The model is compiled using the Adam optimizer and categorical cross entropy is the preferred loss function as we are dealing with multiple output classes. We, then fit the data on the train set and train it for 10 epochs with a batch size of 128.

Once the model is trained, it is then tested and then deployed onto an ml.m5.xlarge instance.

| Architecture

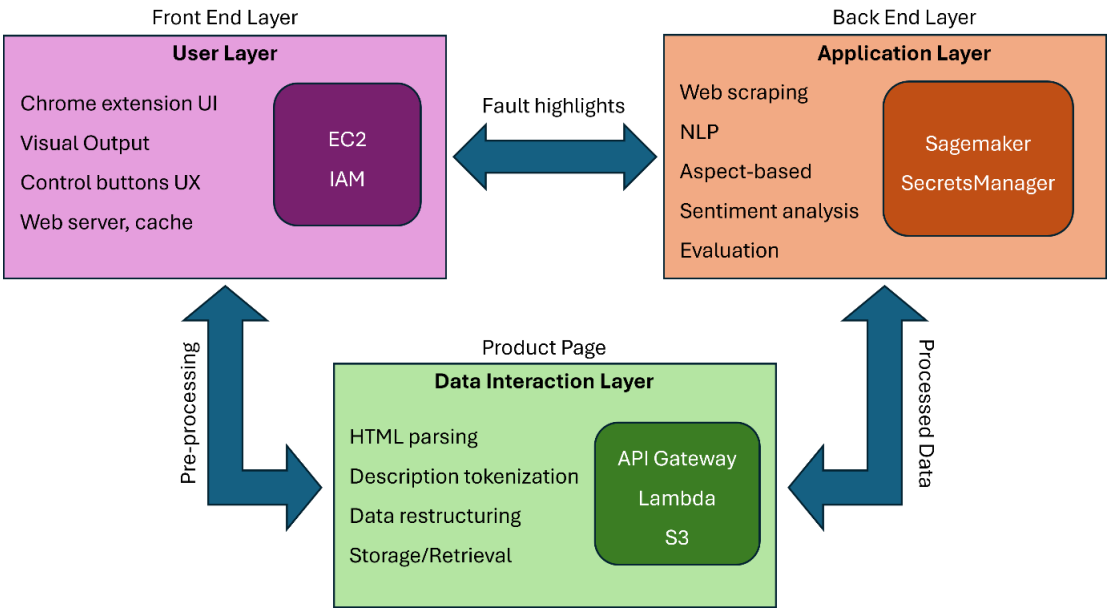


Figure 1: Architecture Layout

Overview of System Layers

The architecture uses a tri-layered model that utilizes modularity to allow for distribution of functionality. The three primary layers are User Layer, Application Layer, and Data Interaction Layer. Each layer is designed to handle relevant tasks, improving scalability and performance.

1. Front End Layer (User Layer)

The User Layer focuses on providing an intuitive interface for users and ensures seamless interaction with the system. This includes:

- **Chrome Extension UI**: The main point of interaction for users to analyze product descriptions. Users can activate functionalities, view results, and get visual insights directly on product pages.

- **Visual Output**: Displays the processed results, highlighting mismatches or flaws in the product description.
- **Control Buttons and UX**: Interactive controls allow users to initiate analysis and customize settings.
- **Web Server & Cache**:
    - **AWS EC2**: Hosts the backend of the Chrome extension, providing computing power and caching for low-latency responses.
    - **AWS IAM**: Manages access control, ensuring secure communication between different AWS services.

2. Back End Layer (Application Layer)

The Application Layer is responsible for the core computational tasks such as data analysis and NLP model inference. It includes:

- **Web Scraping**: Extracts product descriptions and customer reviews from the e-commerce platform.
- **NLP and Sentiment Analysis**: Processes the extracted data to identify aspects, perform sentiment analysis, and compare customer sentiment with the product description.
- **Evaluation**: Determines whether the product description aligns with customer feedback.
- **AWS SageMaker**:
    - Used for training, fine-tuning, and hosting the NLP model that performs tasks such as aspect-based sentiment analysis.
- **AWS Secrets Manager**:
    - Securely manages sensitive information like API keys and credentials required for accessing web scraping APIs and AWS resources.

3. Data Interaction Layer

This Layer acts as a bridge to fill the gap between raw data and the application layer. It prepares data for analysis and manages communication with the user interface. This layer includes:

- **HTML Parsing**: Extracts the necessary textual data (product descriptions and reviews) from the e-commerce product page.
- **Description Tokenization**: Splits the text into tokens for easier processing by the NLP model.
- **Data Restructuring**: Converts unstructured data into a structured format for further analysis.
- **Storage and Retrieval**:
    - **AWS API Gateway**: Facilitates the communication between the Chrome extension and backend services by handling API requests and responses.
    - **AWS Lambda**: Implements serverless computing for tasks such as data preprocessing, tokenization, and restructuring. It processes data in real-time

- **AWS S3**: Provides scalable storage for intermediate results, processed datasets, or logs, ensuring persistence and easy retrieval of data.

End - to - End Workflow

1. **User Initiation**: Users interact with the Chrome extension UI to analyse the product description.
2. **Data Collection**: HTML parsing is triggered via the **Data Interaction Layer**, scraping product descriptions and customer reviews.
3. **Preprocessing**: Tokenization and data structuring occur using **AWS Lambda**, with temporary storage in **AWS S3**.
4. **Analysis**:
   - Data is passed to the **Application Layer**, where **AWS SageMaker** hosts the fine-tuned NLP model.
   - The model processes the data, performs sentiment analysis, and evaluates alignment between customer sentiment and the product description.
5. **Results**: The results are sent back to the Chrome extension via **API Gateway** and presented to the user in the UI.
6. **Secure Access**: All interactions are secured by **AWS IAM**, and sensitive data is protected by **AWS Secrets Manager**.

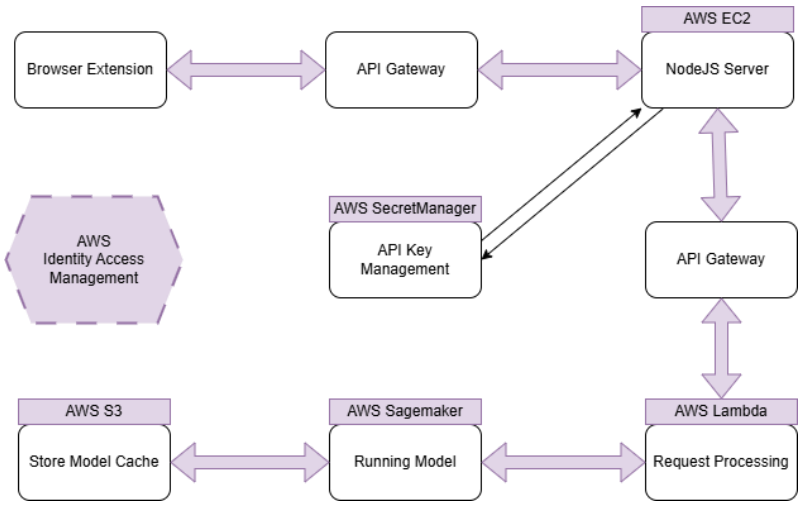


Figure 2: Architecture Workflow

Salient Features

- **Scalability**: The use of **AWS Lambda** and **SageMaker** allows the system to scale efficiently with increased user demand.
- **Cost-Effectiveness**: Serverless computing in **AWS Lambda** ensures cost optimization, while **S3** provides affordable storage options.

- **Security**: **IAM** and **Secrets Manager** ensure secure access and data handling across the architecture.
- **Flexibility**: The modular design allows individual layers to be updated or replaced without affecting the entire system.

| Individual Contribution

Sriram: Web scraping, NLP, Preprocessing, Sentiment analysis

Meghna: Tokenization, Model training, Preprocessing

Navin: Front End, UI/UX, Gateway Integration

Aakash: Backend, Data extraction, Gateway Integration

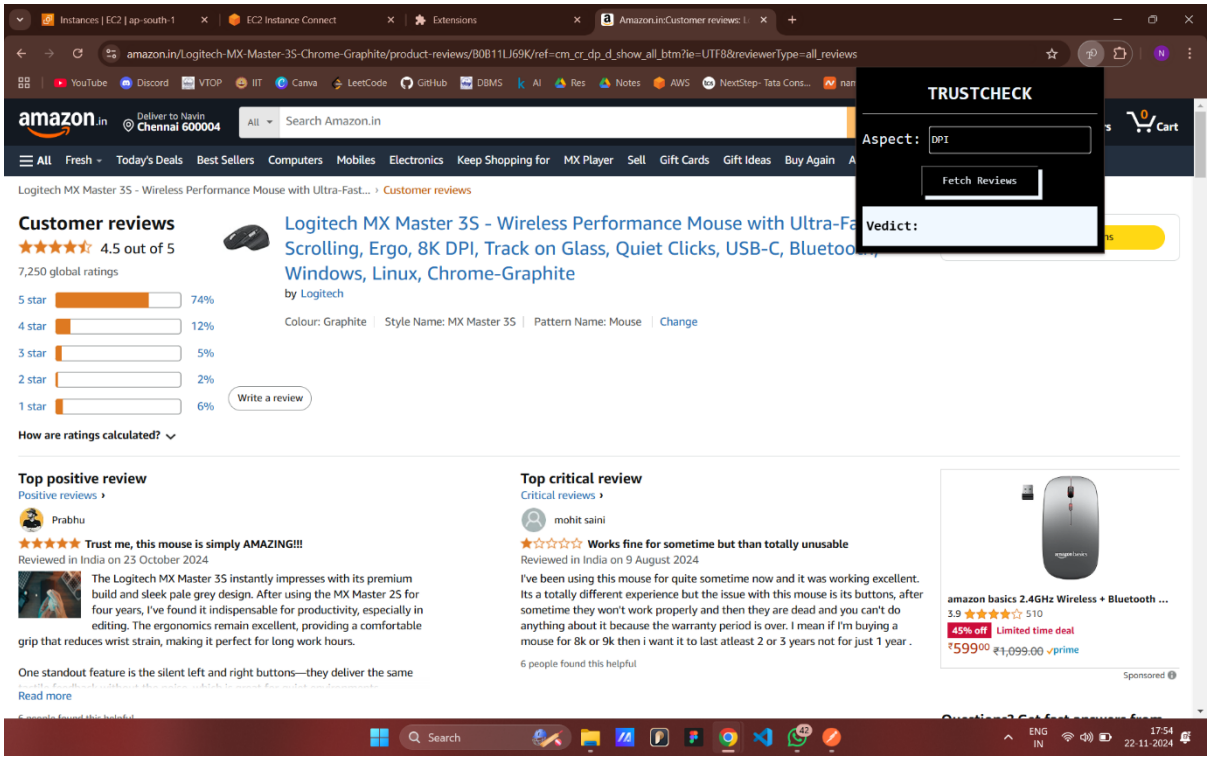
All 4 of us have contributed and committed to the project and have seen it through from start to finish. While the areas of expertise have been mentioned above, when it comes to the using the Amazon Web Services and deploying our project, all 4 of us worked on it together.

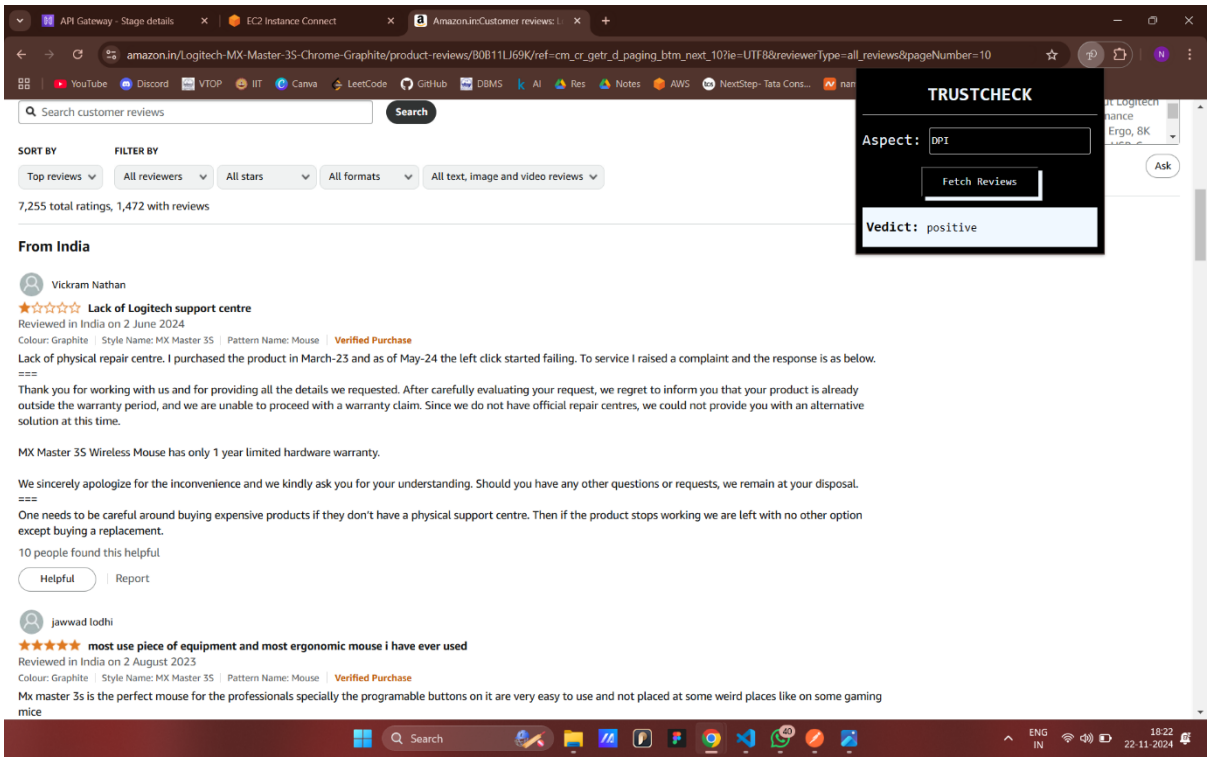
| AWS Services Used

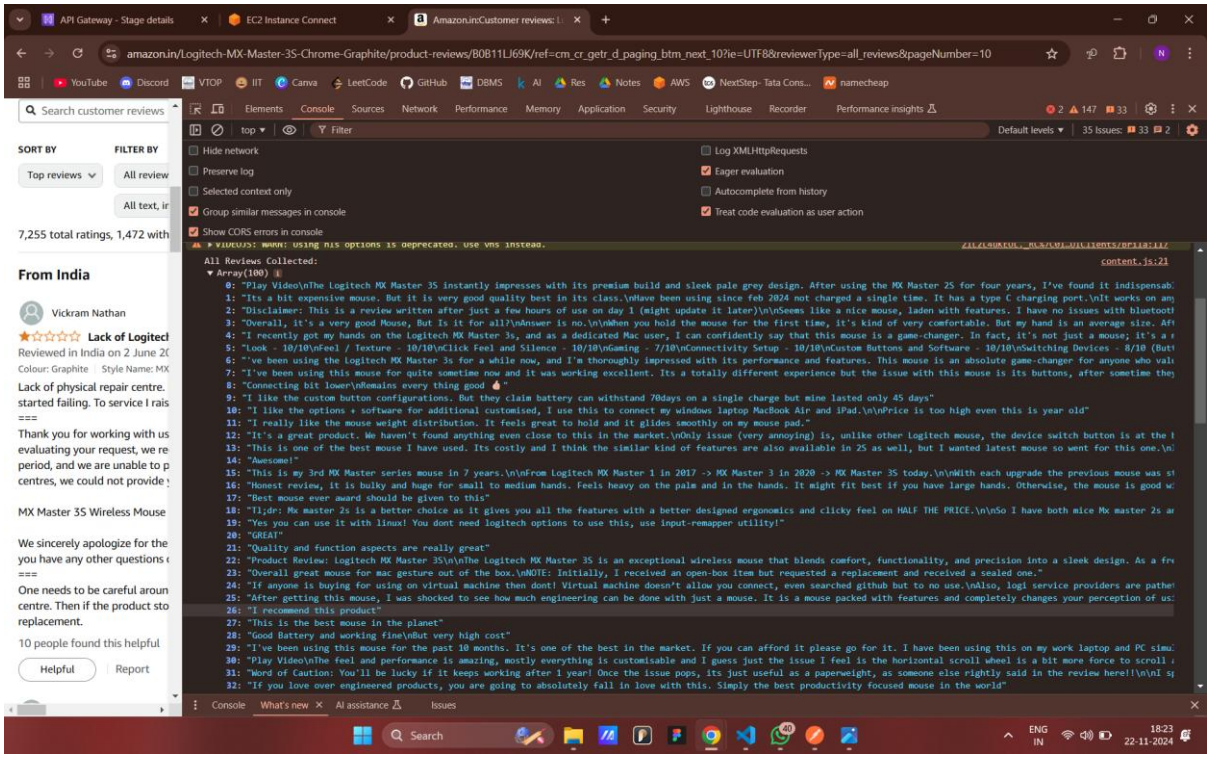
1. **AWS EC2 (Elastic Cloud Compute):** AWS EC2 is used to provide scalable virtual servers in the cloud for running applications and workloads. We are using this service to host our dynamic browser extension (as a web server).
2. **AWS IAM (Identity and Access Management):** Manages user access and permissions in AWS. It helps securely control who can access resources and what actions they can perform, using fine-grained permissions and policies.
3. **AWS API Gateway:** AWS API Gateway is used to create, publish, and manage APIs that allow applications to securely communicate with backend services. We are using this service to create a entry way or access point to the backend services from the front end.
4. **AWS Lambda:** AWS Lambda is used to run code in response to events without provisioning or managing servers, enabling serverless computing. We are using this service as a HTML parser and preprocessor to extract required information.
5. **AWS SageMaker:** AWS SageMaker is used to build, train, and deploy machine learning models at scale in the cloud. We are using this service to train and deploy our own NLP model to perform aspect-based sentiment analysis.
6. **AWS Secrets Manager:** AWS Secrets Manager securely stores and manages sensitive information like API keys and database credentials. It provides automatic rotation, encryption, and easy retrieval through API calls, reducing the risk of exposing secrets in code. It’s ideal for secure and streamlined secret management in AWS environments.
7. **AWS S3 (Simple Storage Service):** AWS S3 (Simple Storage Service) is a scalable cloud storage solution for storing and retrieving data. It’s ideal for backups, data archiving, and serving files, offering secure, durable, and highly available storage in the cloud.

| Output Screenshots

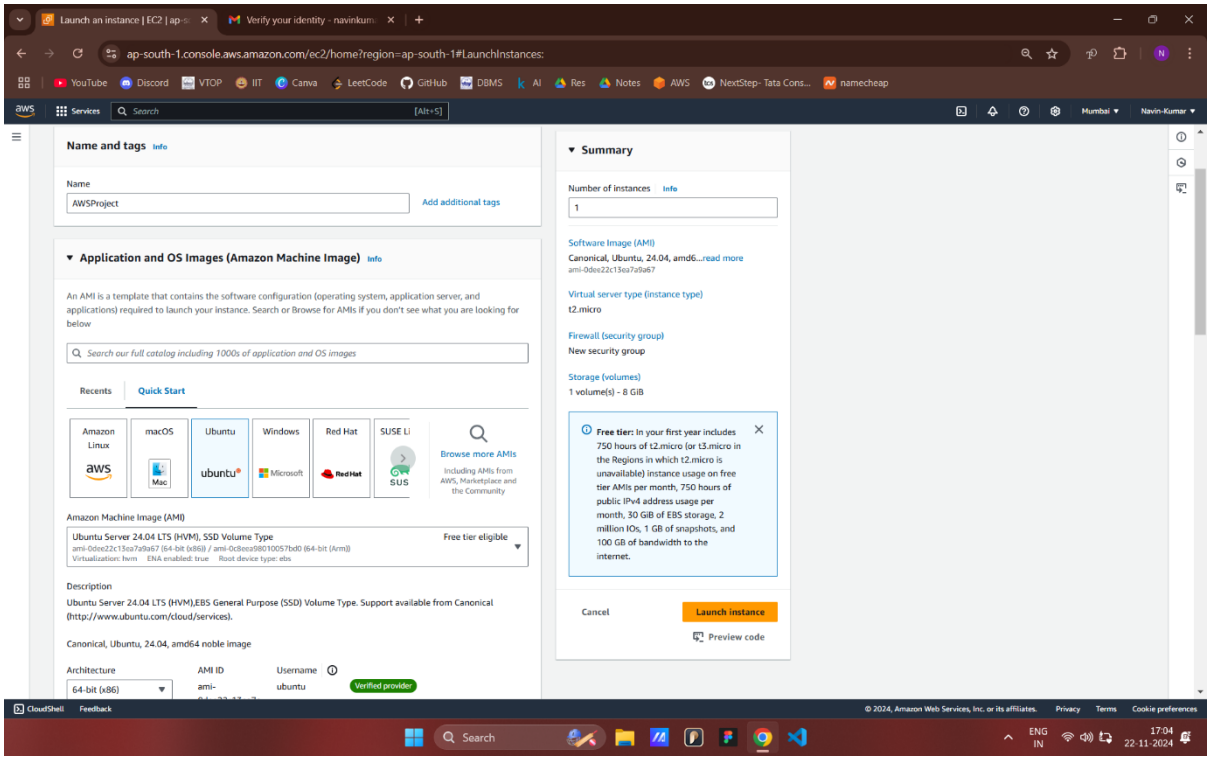
- Browser Extension
    - UI of the extension

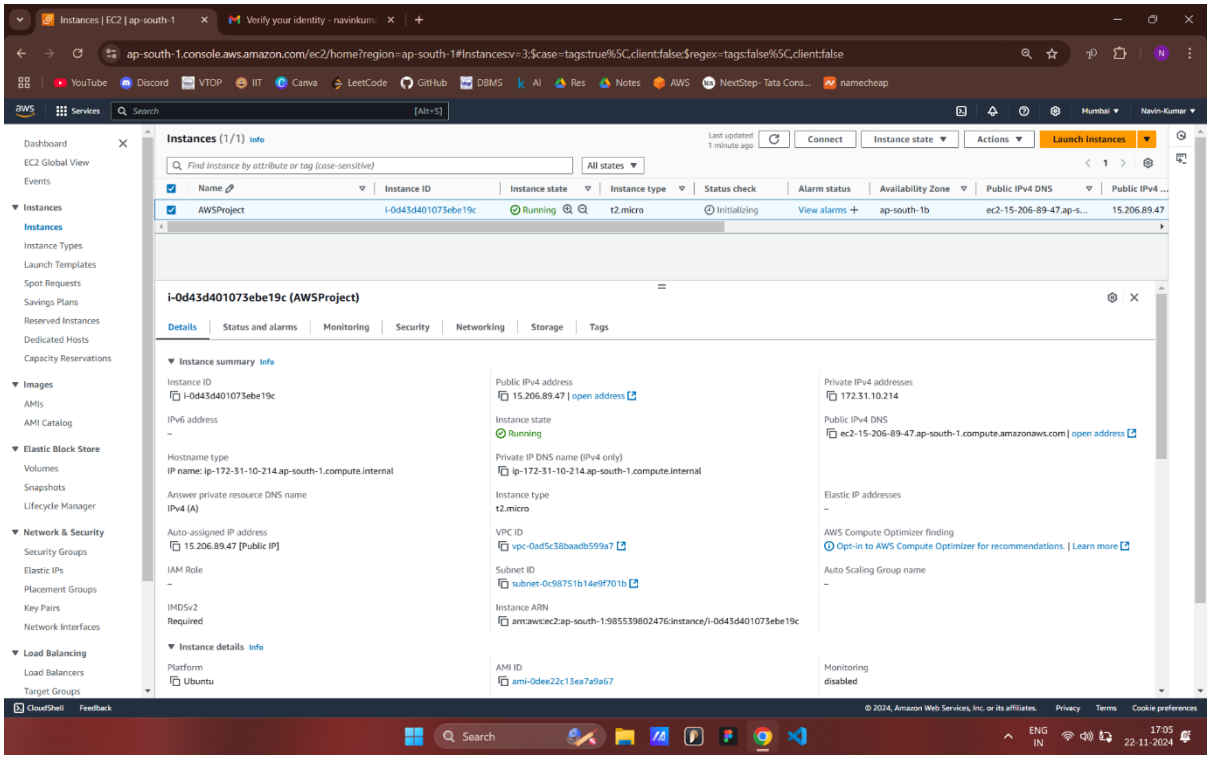


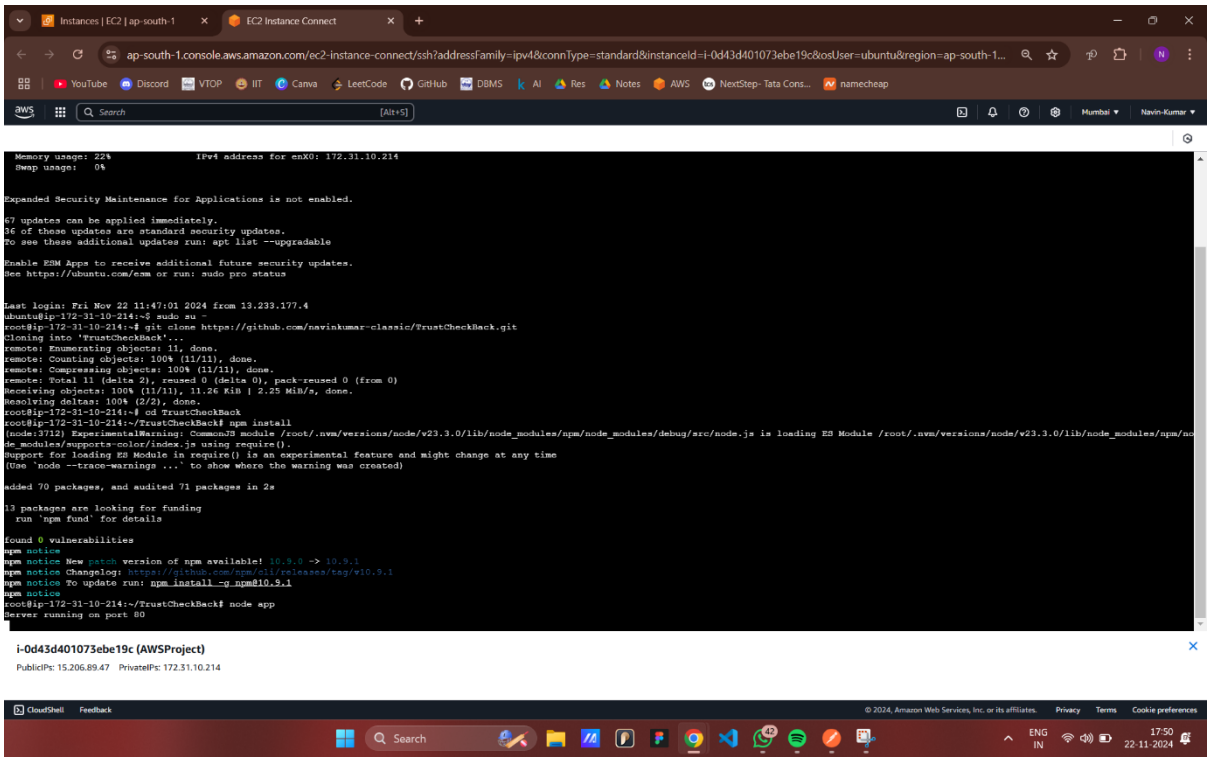


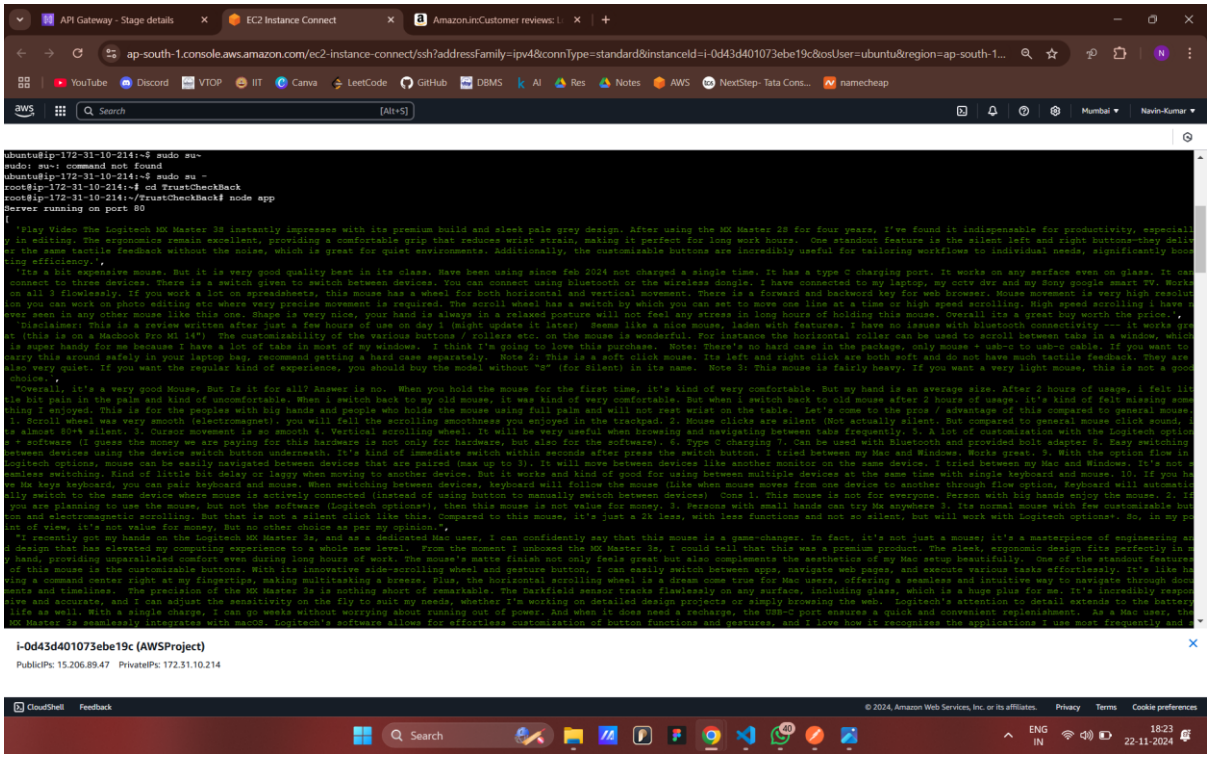


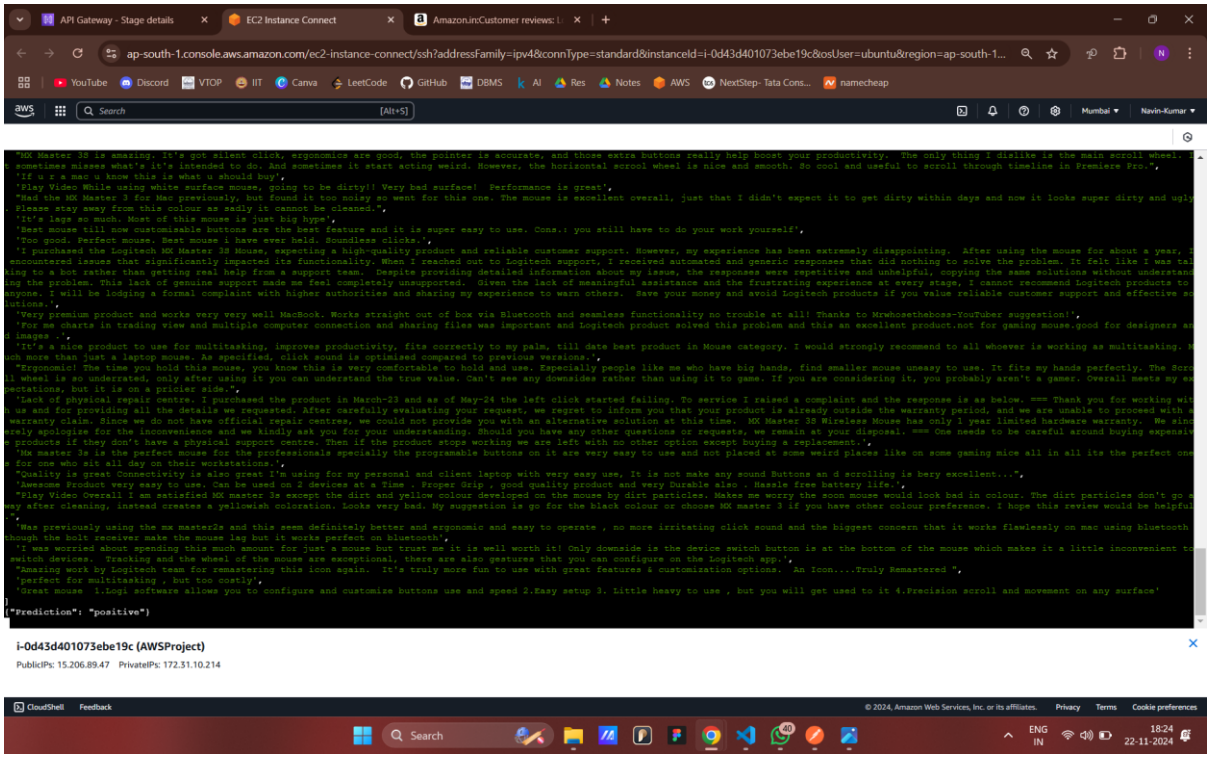
- Backend Server
    - EC2



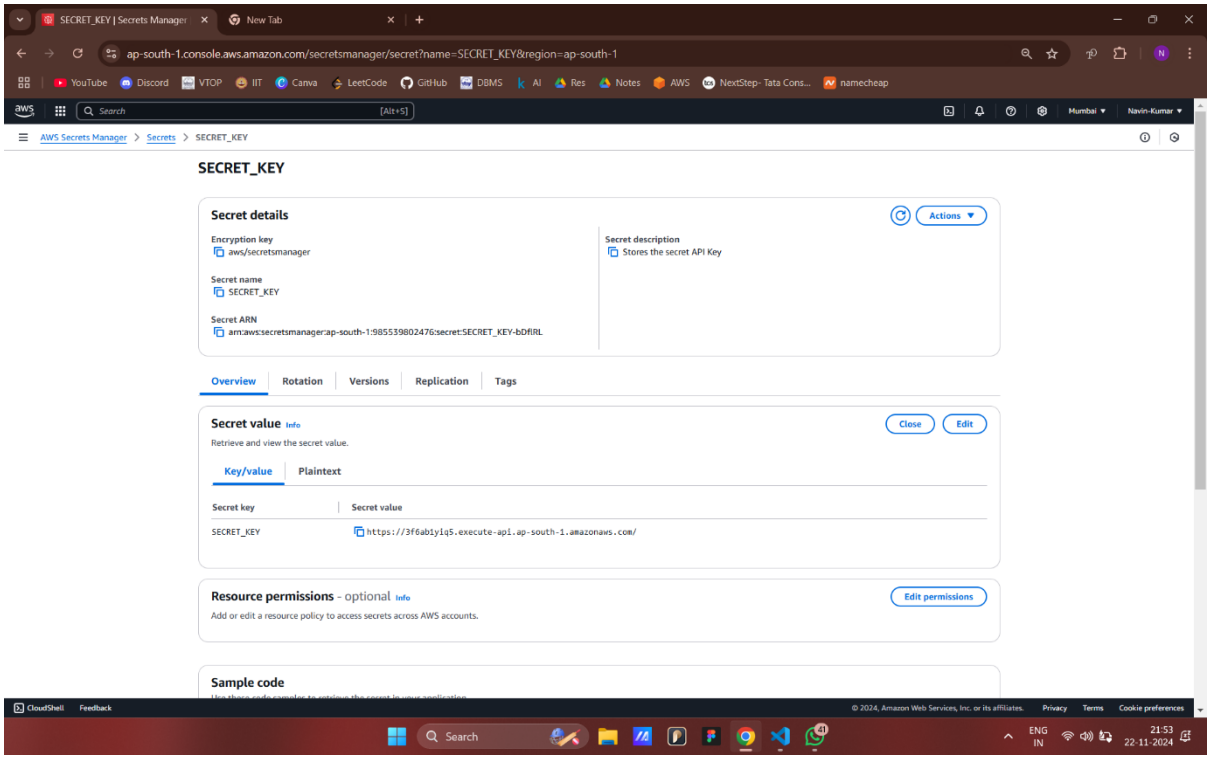

Instances | EC2 | ap-south-1
Verify your identity - navinkum
ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#Instances:v=3;$case=tags:true%5C,client:false;$regex=tags:false%5C,client:false
Instances (1/1) Info
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
All states
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
Public IPv4 ...
AWSProject
i-0d43d401073ebe19c
Running
t2.micro
Initializing
View alarms
ap-south-1b
ec2-15-206-89-47.ap-s...
15.206.89.47
i-0d43d401073ebe19c (AWSProject)
Details
Status and alarms
Monitoring
Security
Networking
Storage
Tags
Instance summary Info
Instance ID
i-0d43d401073ebe19c
Public IPv4 address
15.206.89.47 | open address
Private IPv4 addresses
172.31.10.214
IPv6 address
Instance state
Running
Public IPv4 DNS
ec2-15-206-89-47.ap-south-1.compute.amazonaws.com | open address
Hostname type
IP name: ip-172-31-10-214.ap-south-1.compute.internal
Private IP DNS name (IPv4 only)
ip-172-31-10-214.ap-south-1.compute.internal
Answer private resource DNS name
IPv4 (A)
Instance type
t2.micro
Elastic IP addresses
Auto-assigned IP address
15.206.89.47 [Public IP]
VPC ID
vpc-0ad5c38baadb599a7
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role
Subnet ID
subnet-0c98751b14e9f701b
Auto Scaling Group name
IMDSv2
Required
Instance ARN
arn:aws:ec2:ap-south-1:985539802476:instance/i-0d43d401073ebe19c
Instance details Info
Platform
Ubuntu
AMI ID
ami-0dee22c13ea7a9a67
Monitoring
disabled
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
17:05
22-11-2024


Instances | EC2 | ap-south-1
EC2 Instance Connect
ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?addressFamily=ipv4&connType=standard&instanceId=i-0d43d401073ebe19c&osUser=ubuntu&region=ap-south-1...
Memory usage: 22%
IPv4 address for enX0: 172.31.10.214
Swap usage: 0%
Expanded Security Maintenance for Applications is not enabled.
67 updates can be applied immediately.
36 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status
Last login: Fri Nov 22 11:47:01 2024 from 13.233.177.4
ubuntu@ip-172-31-10-214:~$ sudo su -
root@ip-172-31-10-214:~$ git clone https://github.com/navinkumar-classic/TrustCheckBack.git
Cloning into 'TrustCheckBack'...
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 11 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (11/11), 11.26 KiB | 2.25 MiB/s, done.
Resolving deltas: 100% (2/2), done.
root@ip-172-31-10-214:~$ cd TrustCheckBack
root@ip-172-31-10-214:~/TrustCheckBack$ npm install
(node:3712) ExperimentalWarning: CommonJS module /root/.nvm/versions/node/v23.3.0/lib/node_modules/npm/node_modules/debug/src/node.js is loading ES Module /root/.nvm/versions/node/v23.3.0/lib/node_modules/npm/node_modules/supports-color/index.js using require().
Support for loading ES Module in require() is an experimental feature and might change at any time
(Use `node --trace-warnings ...` to show where the warning was created)
added 70 packages, and audited 71 packages in 2s
13 packages are looking for funding
run `npm fund` for details
found 0 vulnerabilities
npm notice
npm notice New patch version of npm available! 10.9.0 -> 10.9.1
npm notice Changelog: https://github.com/npm/cli/releases/tag/v10.9.1
npm notice To update run: npm install -g npm@10.9.1
npm notice
root@ip-172-31-10-214:~/TrustCheckBack$ node app
Server running on port 80
i-0d43d401073ebe19c (AWSProject)
PublicIPs: 15.206.89.47 PrivateIPs: 172.31.10.214
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
17:50
22-11-2024

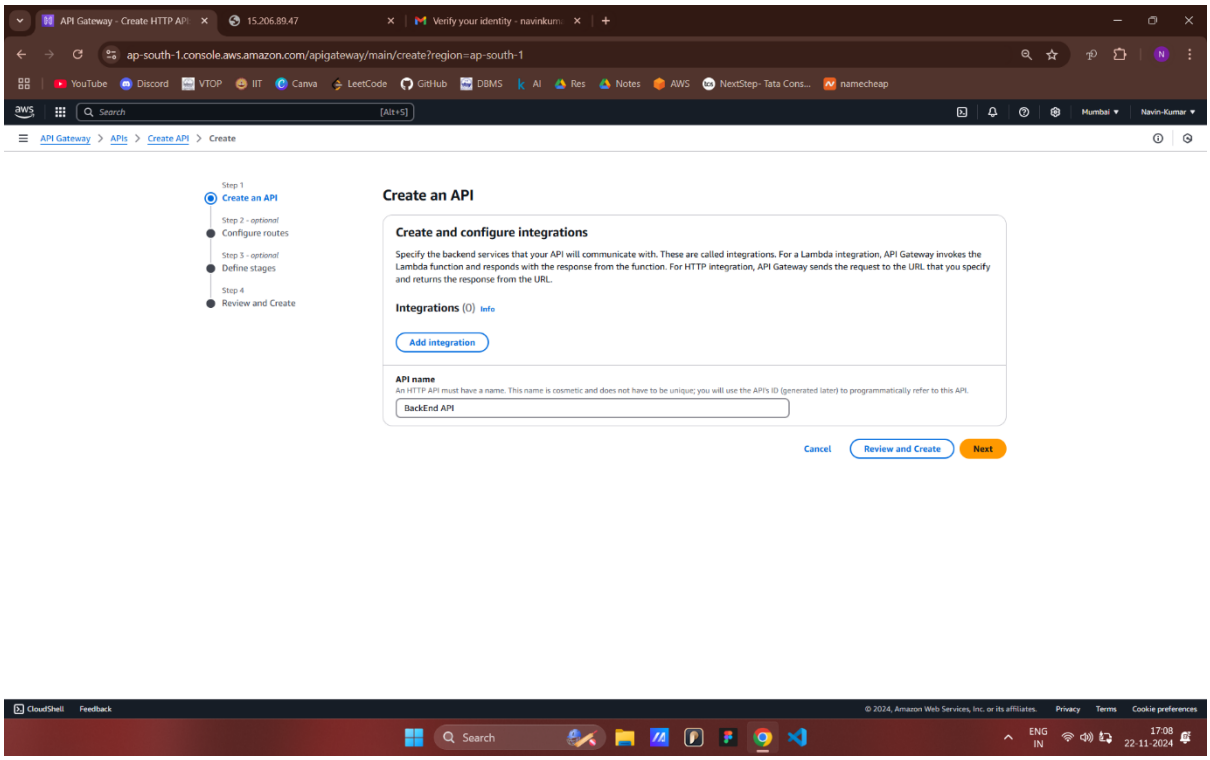


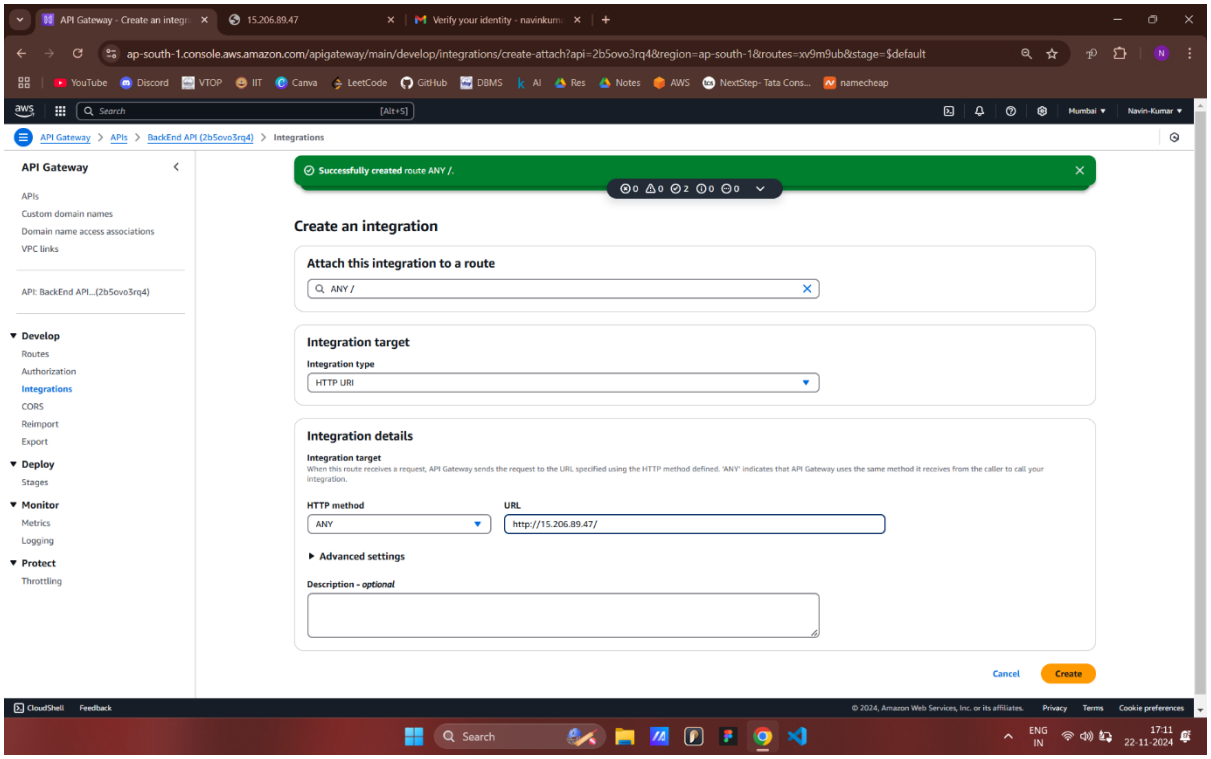


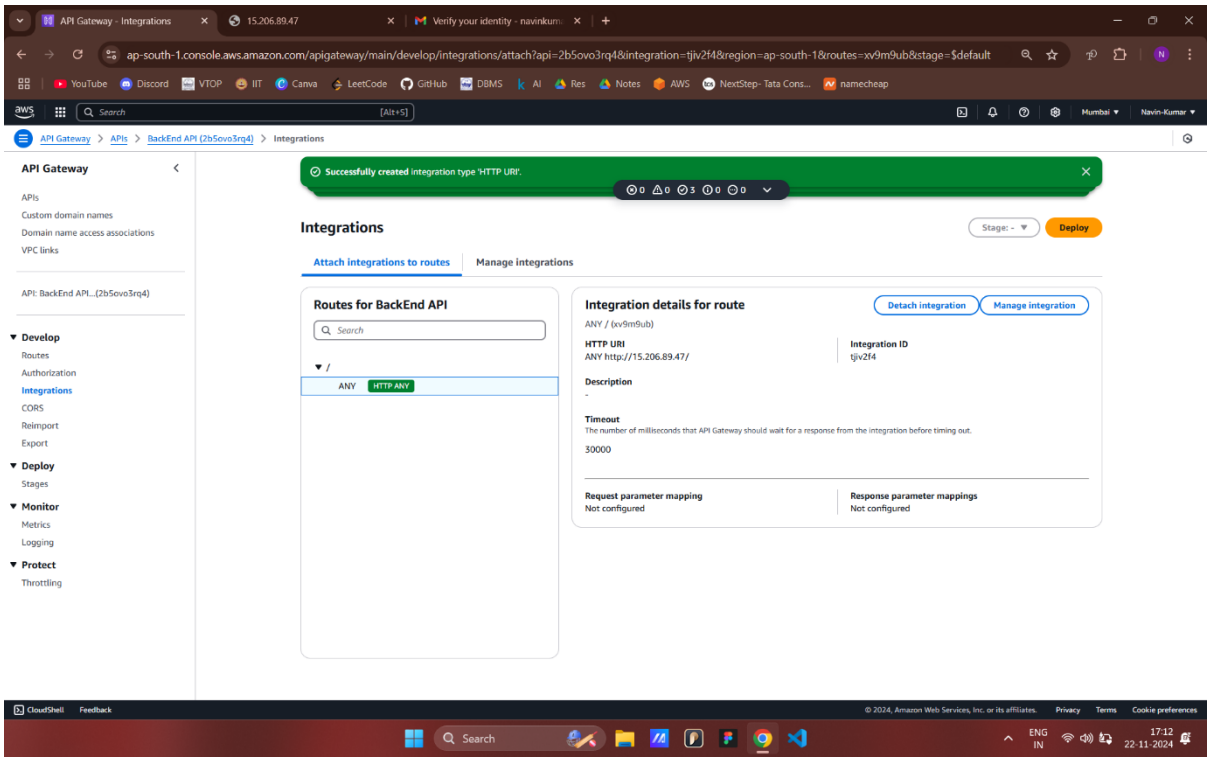
- AWS Secret Manager

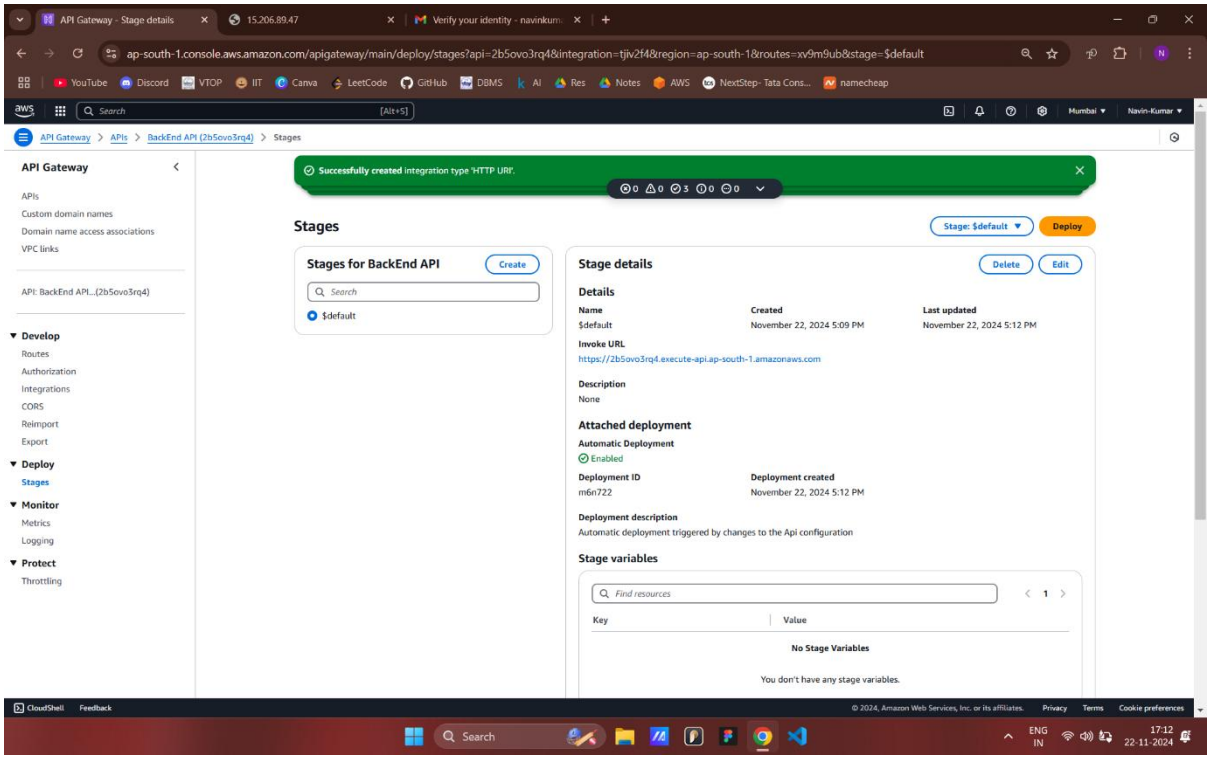


- AWS API Gateway

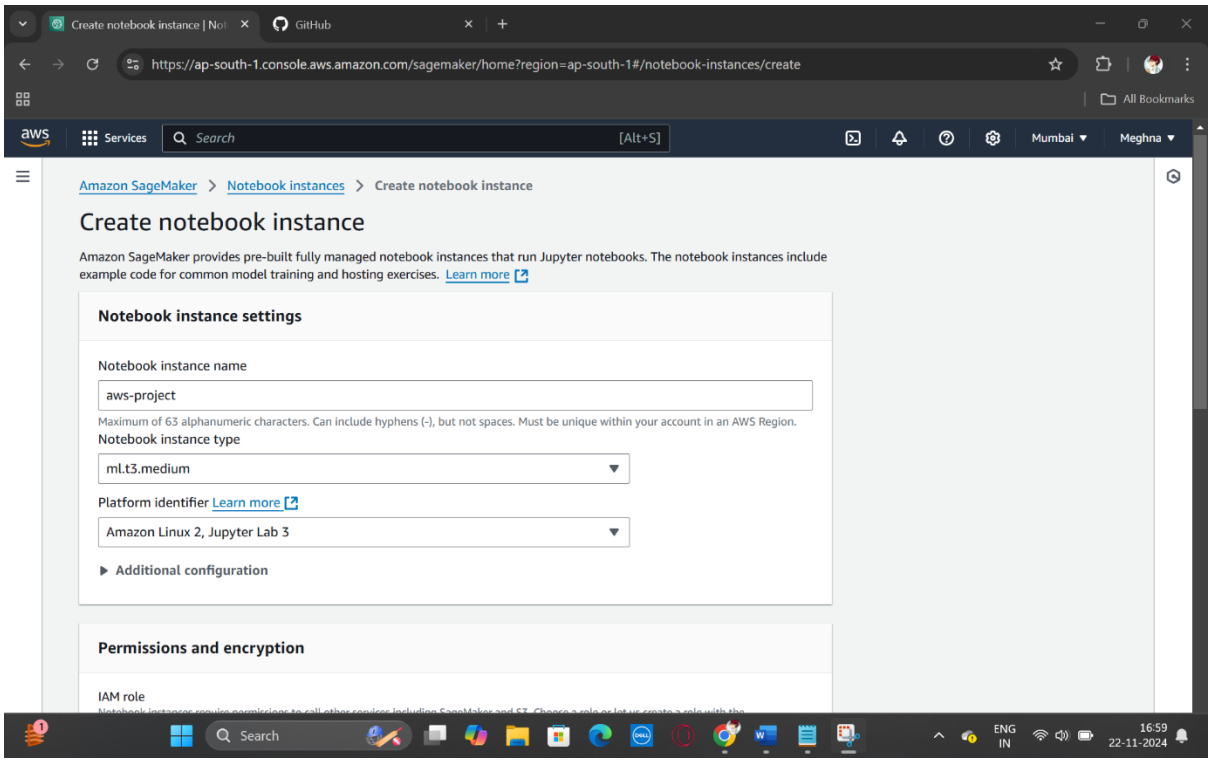


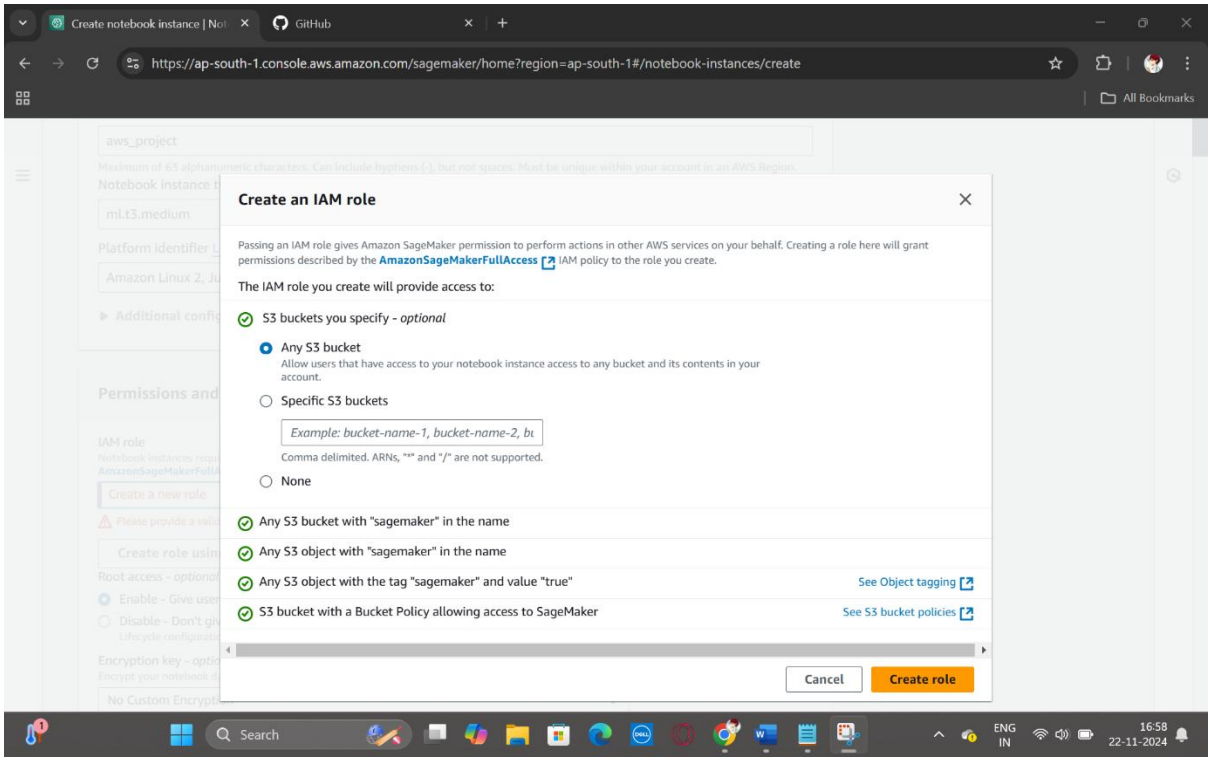

API Gateway - Create an integr
15.206.89.47
Verify your identity - navinkum
ap-south-1.console.aws.amazon.com/apigateway/main/develop/integrations/create-attach?api=2b5ovo3rq4&region=ap-south-1&routes=xv9m9ub&stage=$default
API Gateway > APIs > BackEnd API (2b5ovo3rq4) > Integrations
Successfully created route ANY /.
Create an integration
Attach this integration to a route
ANY /
Integration target
Integration type
HTTP URI
Integration details
Integration target
When this route receives a request, API Gateway sends the request to the URL specified using the HTTP method defined. 'ANY' indicates that API Gateway uses the same method it receives from the caller to call your integration.
HTTP method
ANY
URL
http://15.206.89.47/
Advanced settings
Description - optional
Cancel
Create
API Gateway
APIs
Custom domain names
Domain name access associations
VPC links
API: BackEnd API...(2b5ovo3rq4)
Develop
Routes
Authorization
Integrations
CORS
Reimport
Export
Deploy
Stages
Monitor
Metrics
Logging
Protect
Throttling
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

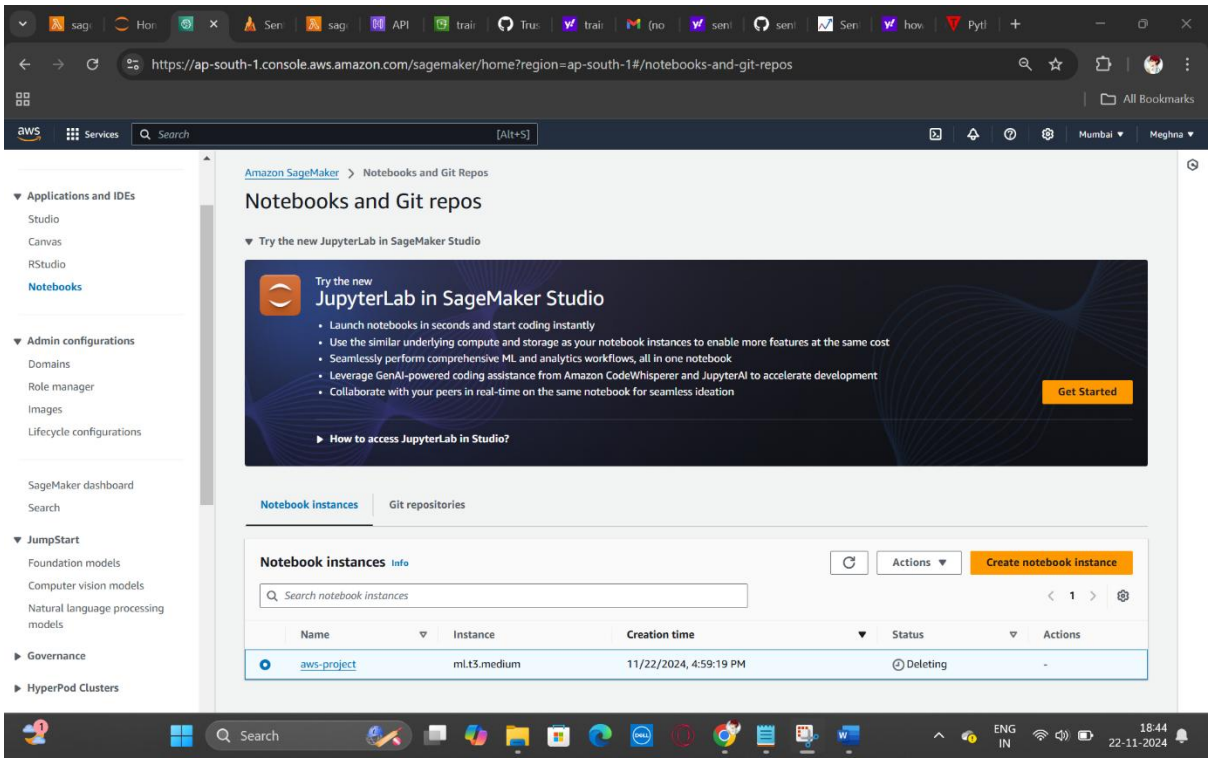

API Gateway - Integrations
15.206.89.47
Verify your identity - navinkum
ap-south-1.console.aws.amazon.com/apigateway/main/develop/integrations/attach?api=2b5ovo3rq4&integration=tjiv2f4&region=ap-south-1&routes=xv9m9ub&stage=$default
API Gateway > APIs > BackEnd API (2b5ovo3rq4) > Integrations
Successfully created integration type 'HTTP URI'.
Integrations
Stage: -
Deploy
Attach integrations to routes
Manage integrations
Routes for BackEnd API
Search
/
ANY
HTTP ANY
Integration details for route
Detach integration
Manage integration
ANY / (xv9m9ub)
HTTP URI
ANY http://15.206.89.47/
Integration ID
tjiv2f4
Description
-
Timeout
The number of milliseconds that API Gateway should wait for a response from the integration before timing out.
30000
Request parameter mapping
Not configured
Response parameter mappings
Not configured
API Gateway
APIs
Custom domain names
Domain name access associations
VPC links
API: BackEnd API...(2b5ovo3rq4)
Develop
Routes
Authorization
Integrations
CORS
Reimport
Export
Deploy
Stages
Monitor
Metrics
Logging
Protect
Throttling
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

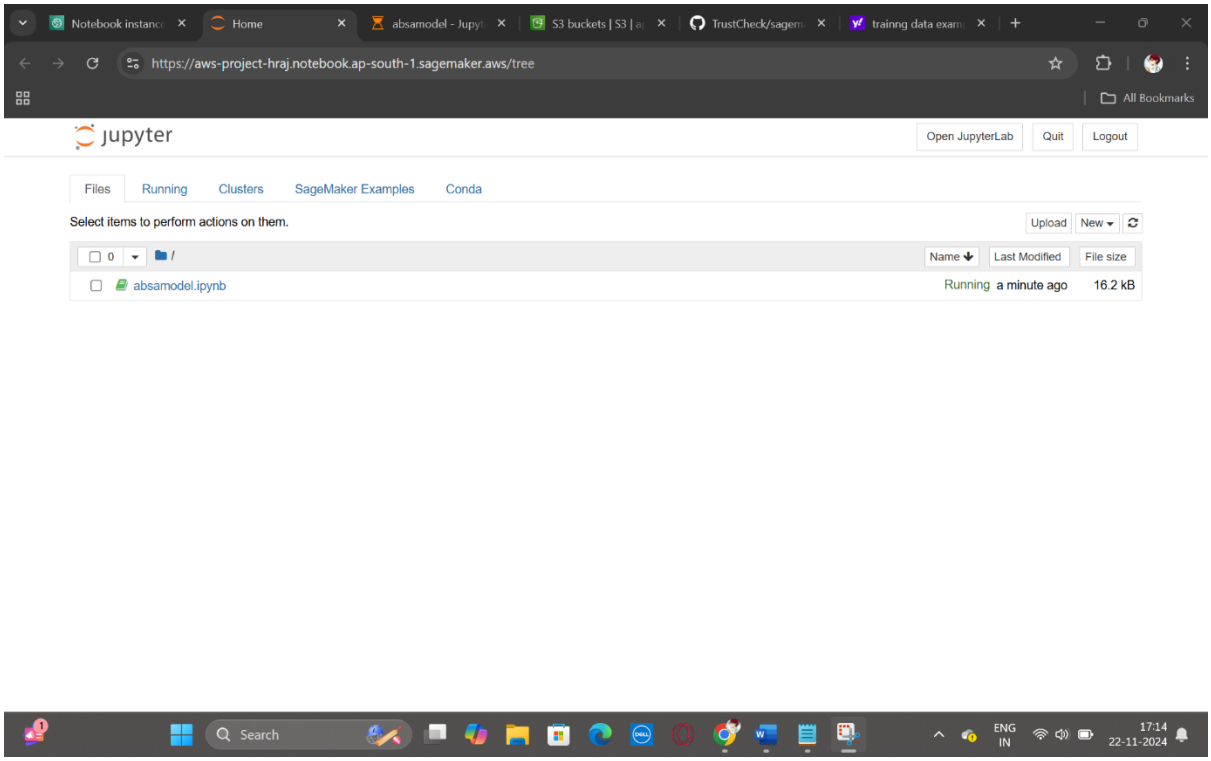

API Gateway - Stage details
ap-south-1.console.aws.amazon.com/apigateway/main/deploy/stages?api=2b5ovo3rq4&integration=tjiv2f4&region=ap-south-1&routes=xv9m9ub&stage=$default
API Gateway
APIs
Custom domain names
Domain name access associations
VPC links
Develop
Routes
Authorization
Integrations
CORS
Reimport
Export
Deploy
Stages
Monitor
Metrics
Logging
Protect
Throttling
Successfully created integration type 'HTTP URI'.
Stages
Stages for BackEnd API
Create
$default
Stage details
Delete
Edit
Details
Name
$default
Created
November 22, 2024 5:09 PM
Last updated
November 22, 2024 5:12 PM
Invoke URL
https://2b5ovo3rq4.execute-api.ap-south-1.amazonaws.com
Description
None
Attached deployment
Automatic Deployment
Enabled
Deployment ID
m6n722
Deployment created
November 22, 2024 5:12 PM
Deployment description
Automatic deployment triggered by changes to the Api configuration
Stage variables
Key
Value
No Stage Variables
You don't have any stage variables.
CloudShell
Feedback

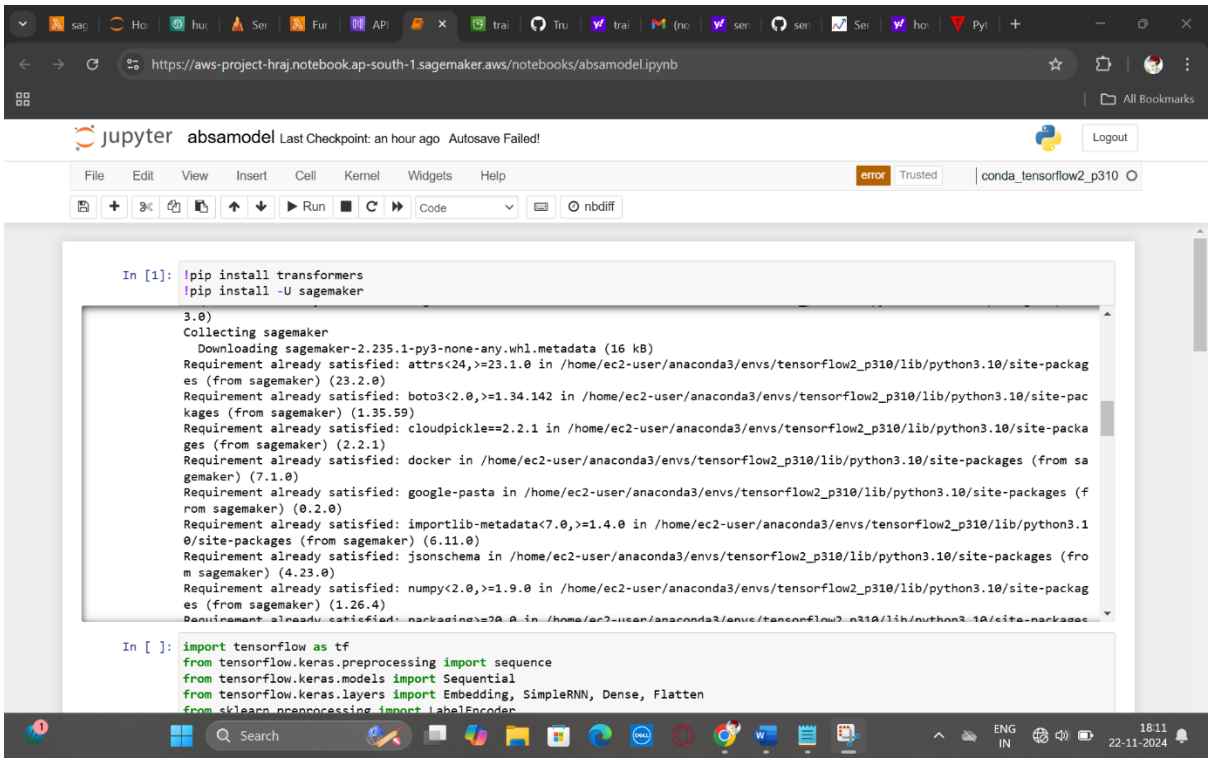
- Sagemaker

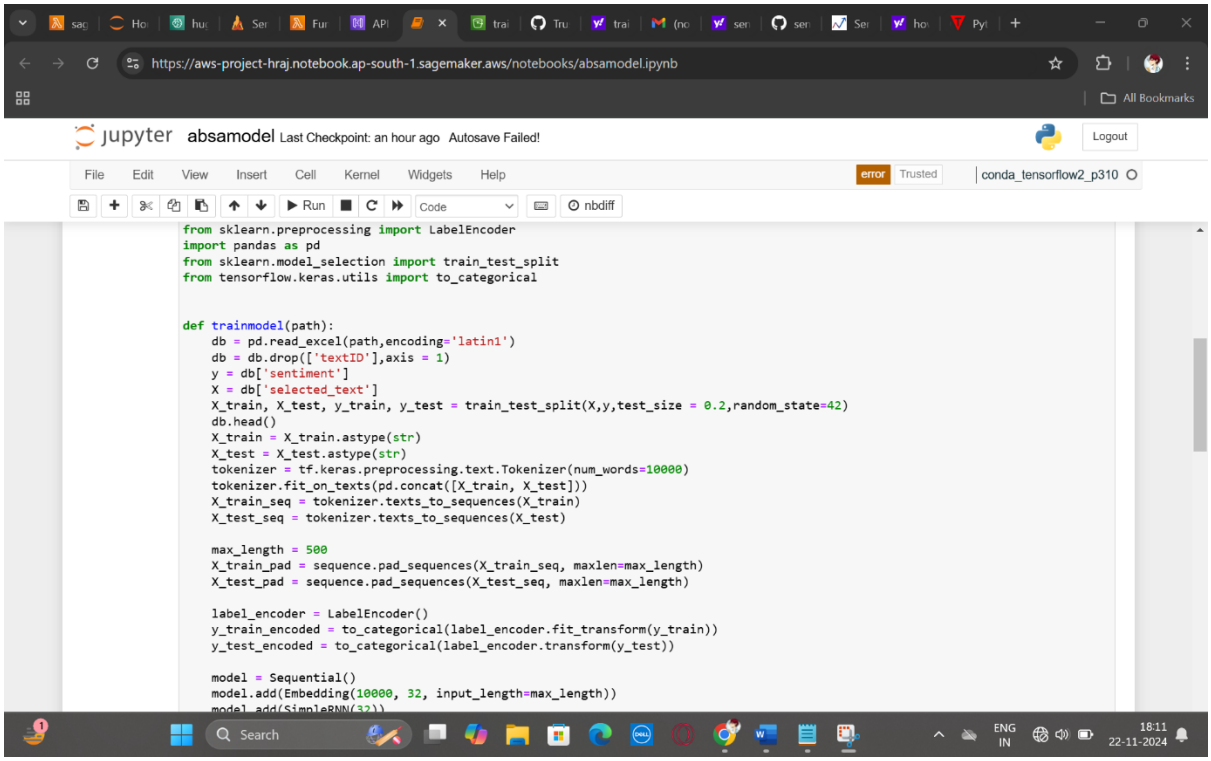

Create notebook instance | Not
GitHub
https://ap-south-1.console.aws.amazon.com/sagemaker/home?region=ap-south-1#/notebook-instances/create
Services
Search
[Alt+S]
Mumbai
Meghna
Amazon SageMaker > Notebook instances > Create notebook instance
Create notebook instance
Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. Learn more
Notebook instance settings
Notebook instance name
aws-project
Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.
Notebook instance type
ml.t3.medium
Platform identifier Learn more
Amazon Linux 2, Jupyter Lab 3
Additional configuration
Permissions and encryption
IAM role

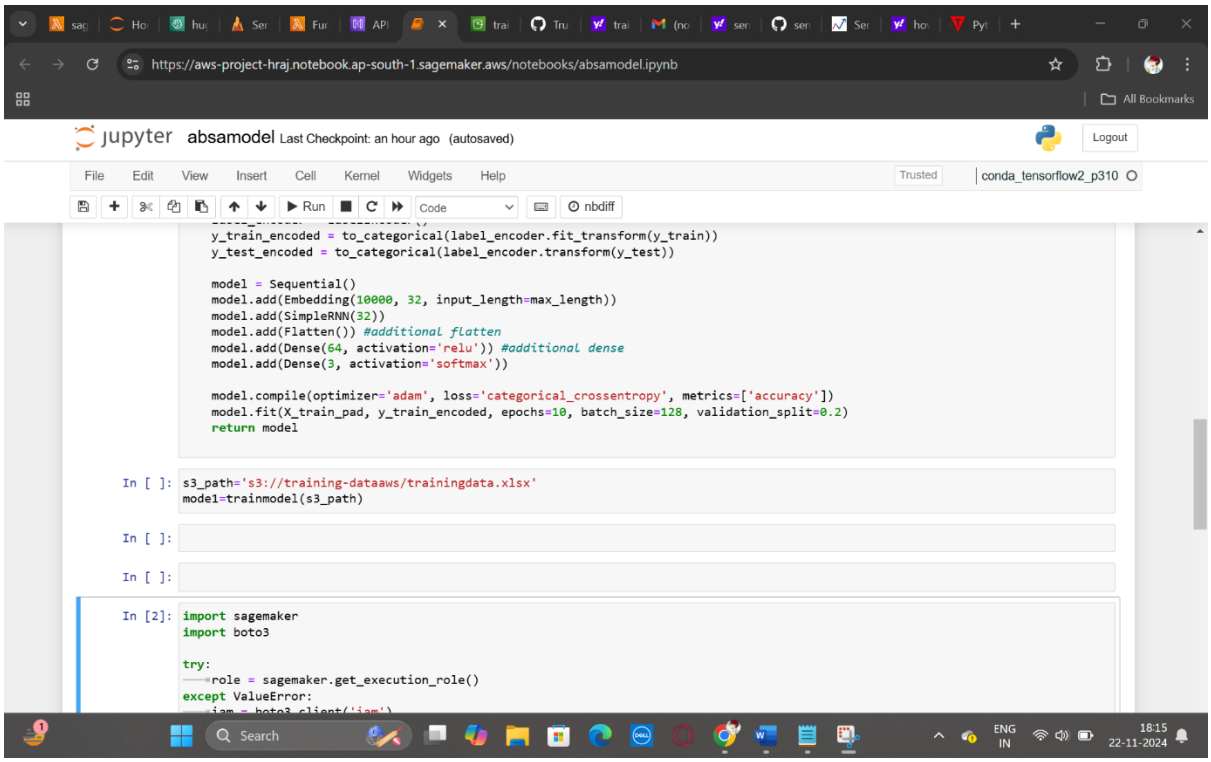

Create notebook instance | Not
GitHub
https://ap-south-1.console.aws.amazon.com/sagemaker/home?region=ap-south-1#/notebook-instances/create
All Bookmarks
Create an IAM role
Passing an IAM role gives Amazon SageMaker permission to perform actions in other AWS services on your behalf. Creating a role here will grant permissions described by the AmazonSageMakerFullAccess IAM policy to the role you create.
The IAM role you create will provide access to:
S3 buckets you specify - optional
Any S3 bucket
Allow users that have access to your notebook instance access to any bucket and its contents in your account.
Specific S3 buckets
Example: bucket-name-1, bucket-name-2, bu
Comma delimited. ARNs, "*" and "/" are not supported.
None
Any S3 bucket with "sagemaker" in the name
Any S3 object with "sagemaker" in the name
Any S3 object with the tag "sagemaker" and value "true"
See Object tagging
S3 bucket with a Bucket Policy allowing access to SageMaker
See S3 bucket policies
Cancel
Create role
Search
ENG
IN
16:58
22-11-2024

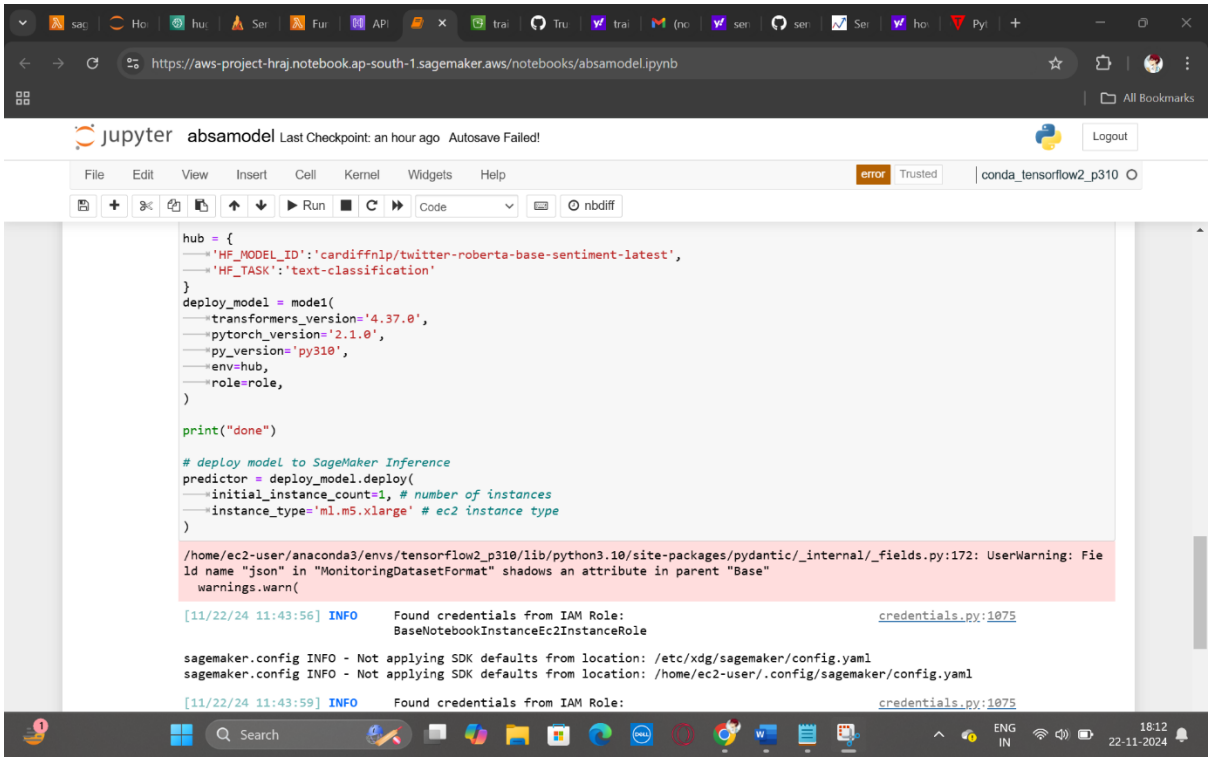

https://ap-south-1.console.aws.amazon.com/sagemaker/home?region=ap-south-1#/notebooks-and-git-repos
All Bookmarks
aws
Services
Search
[Alt+S]
Mumbai
Meghna
Amazon SageMaker > Notebooks and Git Repos
Notebooks and Git repos
Applications and IDEs
Studio
Canvas
RStudio
Notebooks
Admin configurations
Domains
Role manager
Images
Lifecycle configurations
SageMaker dashboard
Search
JumpStart
Foundation models
Computer vision models
Natural language processing models
Governance
HyperPod Clusters
Try the new JupyterLab in SageMaker Studio
Try the new
JupyterLab in SageMaker Studio
Launch notebooks in seconds and start coding instantly
Use the similar underlying compute and storage as your notebook instances to enable more features at the same cost
Seamlessly perform comprehensive ML and analytics workflows, all in one notebook
Leverage GenAI-powered coding assistance from Amazon CodeWhisperer and JupyterAI to accelerate development
Collaborate with your peers in real-time on the same notebook for seamless ideation
Get Started
How to access JupyterLab in Studio?
Notebook instances
Git repositories
Notebook instances Info
Actions
Create notebook instance
Search notebook instances
Name
Instance
Creation time
Status
Actions
aws-project
ml.t3.medium
11/22/2024, 4:59:19 PM
Deleting
Search
ENG
IN
18:44
22-11-2024

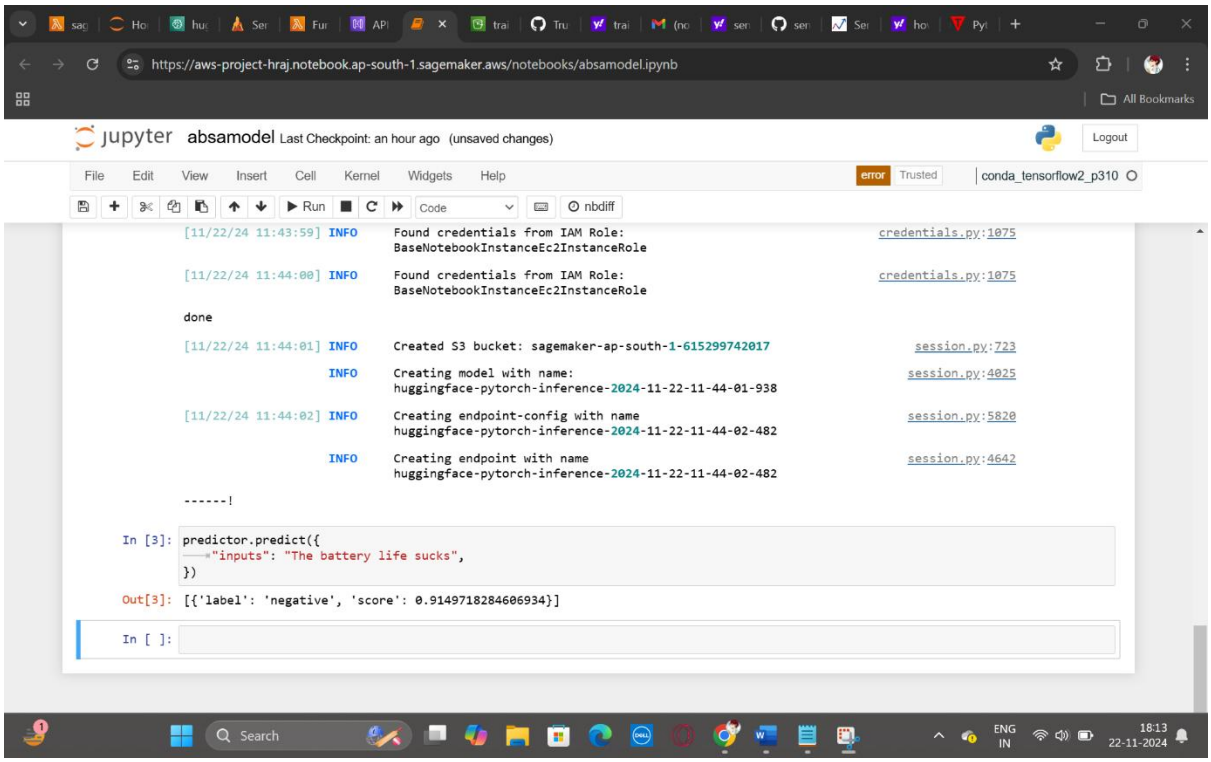

Notebook instanc
Home
absamodel - Jupyt
S3 buckets | S3 | a
TrustCheck/sagem
trainng data exam
https://aws-project-hraj.notebook.ap-south-1.sagemaker.aws/tree
All Bookmarks
jupyter
Open JupyterLab
Quit
Logout
Files
Running
Clusters
SageMaker Examples
Conda
Select items to perform actions on them.
Upload
New
0
/
Name
Last Modified
File size
absamodel.ipynb
Running a minute ago
16.2 kB
Search
ENG
IN
17:14
22-11-2024

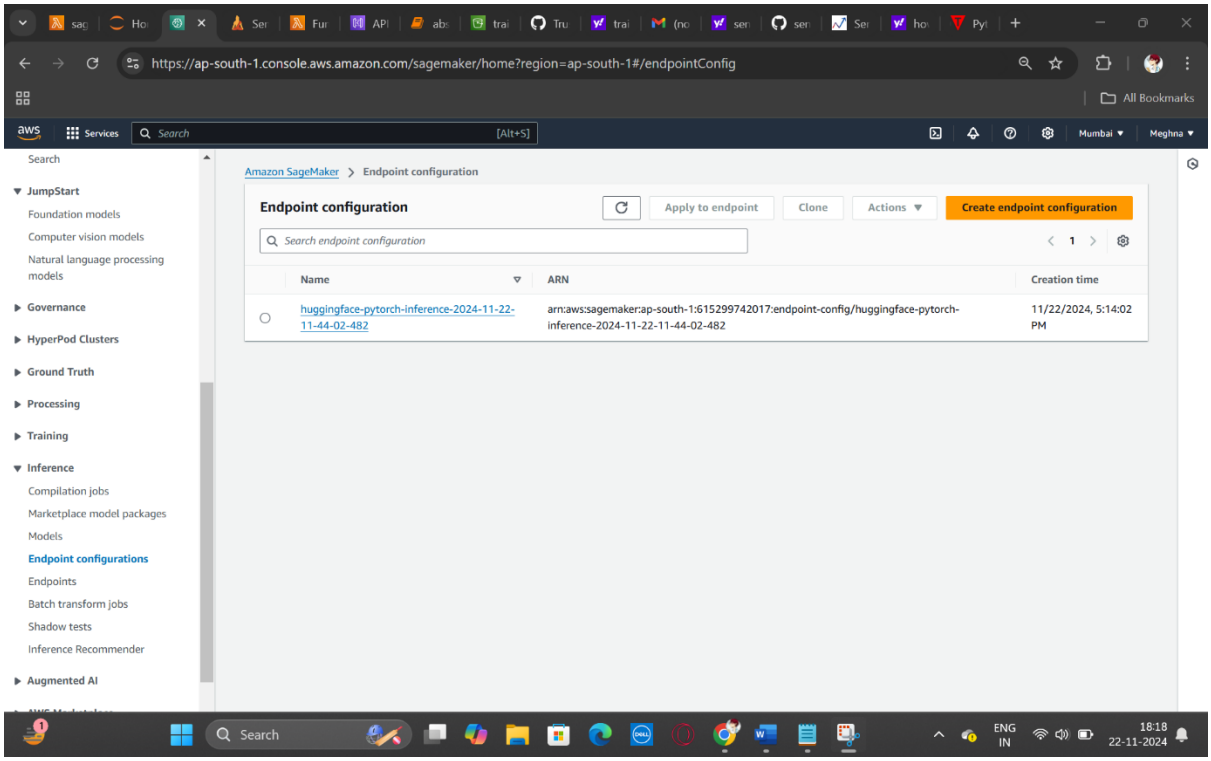

https://aws-project-hraj.notebook.ap-south-1.sagemaker.aws/notebooks/absamodel.ipynb
All Bookmarks
jupyter
absamodel
Last Checkpoint: an hour ago
Autosave Failed!
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
error
Trusted
conda_tensorflow2_p310
Run
Code
nbdiff
In [1]:
!pip install transformers
!pip install -U sagemaker
3.0)
Collecting sagemaker
Downloading sagemaker-2.235.1-py3-none-any.whl.metadata (16 kB)
Requirement already satisfied: attrs<24,>=23.1.0 in /home/ec2-user/anaconda3/envs/tensorflow2_p310/lib/python3.10/site-packages (from sagemaker) (23.2.0)
Requirement already satisfied: boto3<2.0,>=1.34.142 in /home/ec2-user/anaconda3/envs/tensorflow2_p310/lib/python3.10/site-packages (from sagemaker) (1.35.59)
Requirement already satisfied: cloudpickle==2.2.1 in /home/ec2-user/anaconda3/envs/tensorflow2_p310/lib/python3.10/site-packages (from sagemaker) (2.2.1)
Requirement already satisfied: docker in /home/ec2-user/anaconda3/envs/tensorflow2_p310/lib/python3.10/site-packages (from sagemaker) (7.1.0)
Requirement already satisfied: google-pasta in /home/ec2-user/anaconda3/envs/tensorflow2_p310/lib/python3.10/site-packages (from sagemaker) (0.2.0)
Requirement already satisfied: importlib-metadata<7.0,>=1.4.0 in /home/ec2-user/anaconda3/envs/tensorflow2_p310/lib/python3.10/site-packages (from sagemaker) (6.11.0)
Requirement already satisfied: jsonschema in /home/ec2-user/anaconda3/envs/tensorflow2_p310/lib/python3.10/site-packages (from sagemaker) (4.23.0)
Requirement already satisfied: numpy<2.0,>=1.9.0 in /home/ec2-user/anaconda3/envs/tensorflow2_p310/lib/python3.10/site-packages (from sagemaker) (1.26.4)
In [ ]:
import tensorflow as tf
from tensorflow.keras.preprocessing import sequence
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, SimpleRNN, Dense, Flatten
Search
ENG
IN
18:11
22-11-2024

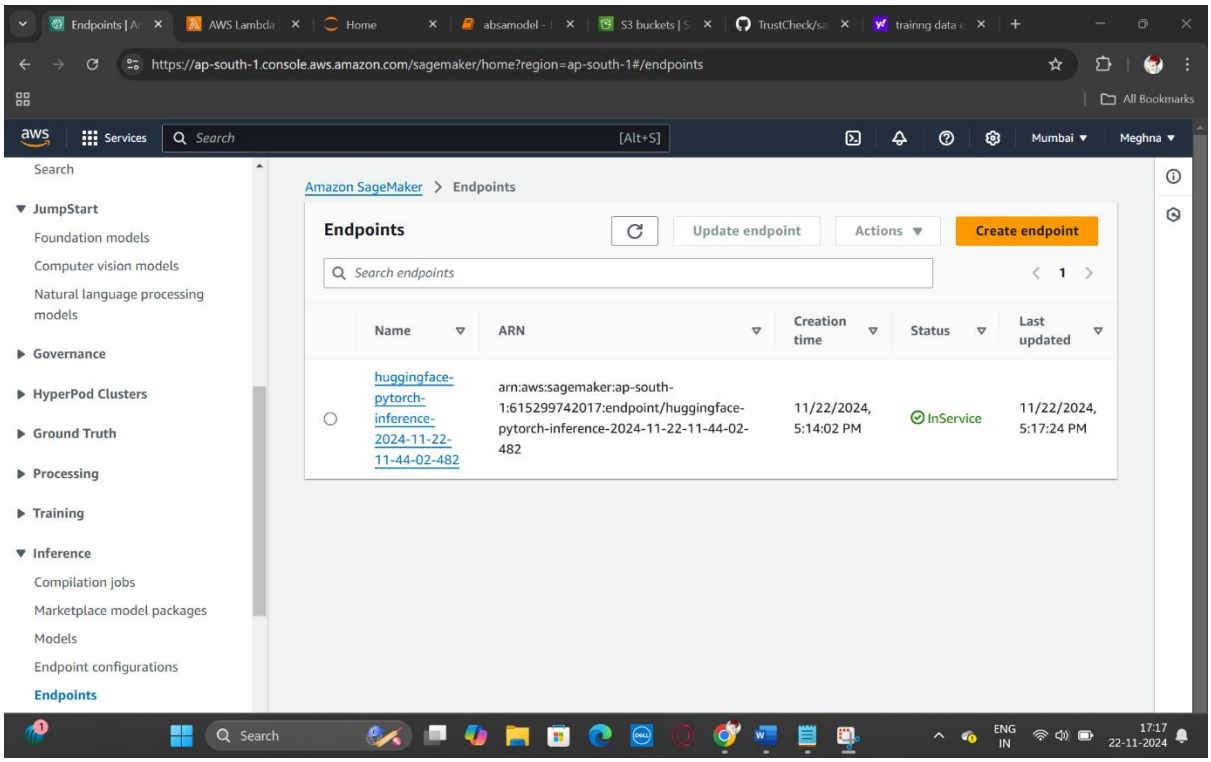

https://aws-project-hraj.notebook.ap-south-1.sagemaker.aws/notebooks/absamodel.ipynb
jupyter absamodel Last Checkpoint: an hour ago Autosave Failed!
Logout
File Edit View Insert Cell Kernel Widgets Help
error Trusted
conda_tensorflow2_p310
Run Code nbdiff
from sklearn.preprocessing import LabelEncoder
import pandas as pd
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import to_categorical
def trainmodel(path):
db = pd.read_excel(path,encoding='latin1')
db = db.drop(['textID'],axis = 1)
y = db['sentiment']
X = db['selected_text']
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.2,random_state=42)
db.head()
X_train = X_train.astype(str)
X_test = X_test.astype(str)
tokenizer = tf.keras.preprocessing.text.Tokenizer(num_words=10000)
tokenizer.fit_on_texts(pd.concat([X_train, X_test]))
X_train_seq = tokenizer.texts_to_sequences(X_train)
X_test_seq = tokenizer.texts_to_sequences(X_test)
max_length = 500
X_train_pad = sequence.pad_sequences(X_train_seq, maxlen=max_length)
X_test_pad = sequence.pad_sequences(X_test_seq, maxlen=max_length)
label_encoder = LabelEncoder()
y_train_encoded = to_categorical(label_encoder.fit_transform(y_train))
y_test_encoded = to_categorical(label_encoder.transform(y_test))
model = Sequential()
model.add(Embedding(10000, 32, input_length=max_length))
Search
ENG IN
18:11 22-11-2024


https://aws-project-hraj.notebook.ap-south-1.sagemaker.aws/notebooks/absamodel.ipynb
jupyter absamodel Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
conda_tensorflow2_p310
Run Code nbdiff
y_train_encoded = to_categorical(label_encoder.fit_transform(y_train))
y_test_encoded = to_categorical(label_encoder.transform(y_test))
model = Sequential()
model.add(Embedding(10000, 32, input_length=max_length))
model.add(SimpleRNN(32))
model.add(Flatten()) #additional flatten
model.add(Dense(64, activation='relu')) #additional dense
model.add(Dense(3, activation='softmax'))
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(X_train_pad, y_train_encoded, epochs=10, batch_size=128, validation_split=0.2)
return model
In [ ]:
s3_path='s3://training-dataaws/trainingdata.xlsx'
model=trainmodel(s3_path)
In [ ]:
In [ ]:
In [2]:
import sagemaker
import boto3
try:
role = sagemaker.get_execution_role()
except ValueError:
Search
ENG IN
18:15 22-11-2024

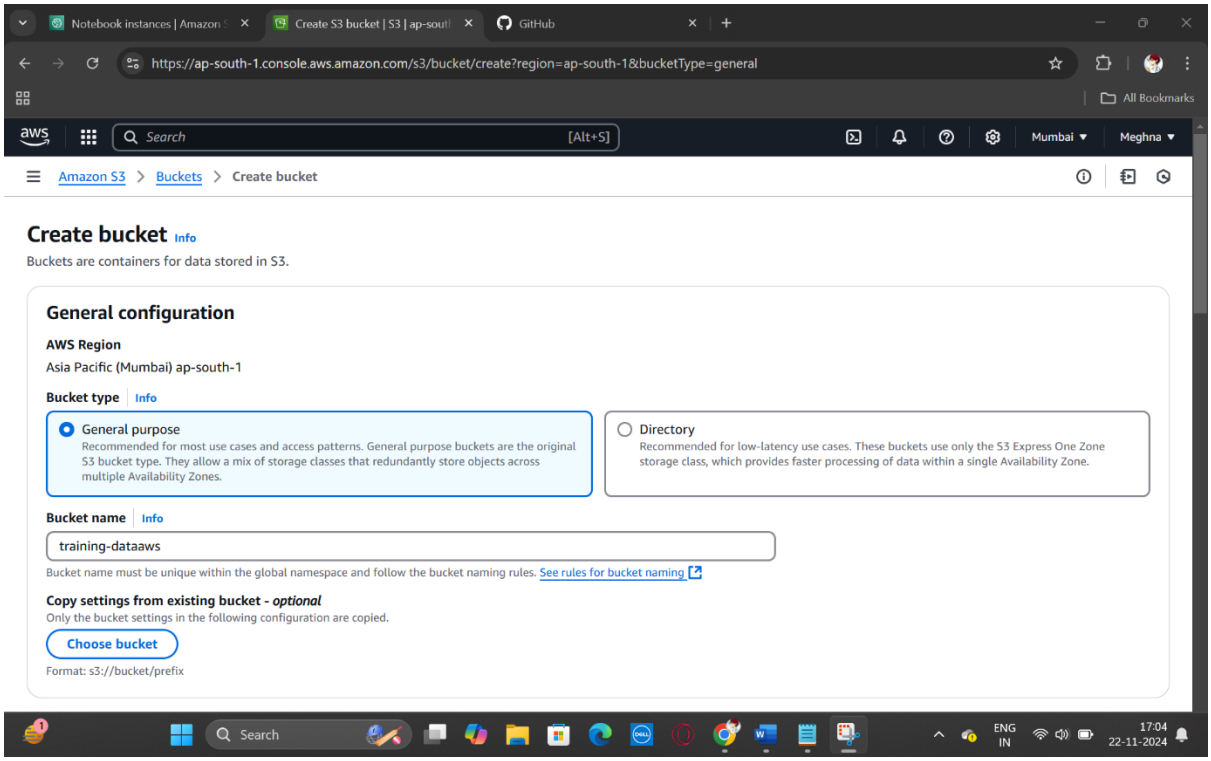

https://aws-project-hraj.notebook.ap-south-1.sagemaker.aws/notebooks/absamodel.ipynb
jupyter absamodel Last Checkpoint: an hour ago Autosave Failed!
Logout
File Edit View Insert Cell Kernel Widgets Help
error Trusted
conda_tensorflow2_p310
Run Code nbdiff
hub = {
'HF_MODEL_ID':'cardiffnlp/twitter-roberta-base-sentiment-latest',
'HF_TASK':'text-classification'
}
deploy_model = model(
transformers_version='4.37.0',
pytorch_version='2.1.0',
py_version='py310',
env=hub,
role=role,
)
print("done")
# deploy model to SageMaker Inference
predictor = deploy_model.deploy(
initial_instance_count=1, # number of instances
instance_type='ml.m5.xlarge' # ec2 instance type
)
/home/ec2-user/anaconda3/envs/tensorflow2_p310/lib/python3.10/site-packages/pydantic/_internal/_fields.py:172: UserWarning: Field name "json" in "MonitoringDatasetFormat" shadows an attribute in parent "Base"
warnings.warn(
[11/22/24 11:43:56] INFO Found credentials from IAM Role: BaseNotebookInstanceEc2InstanceRole credentials.py:1075
sagemaker.config INFO - Not applying SDK defaults from location: /etc/xdg/sagemaker/config.yaml
sagemaker.config INFO - Not applying SDK defaults from location: /home/ec2-user/.config/sagemaker/config.yaml
[11/22/24 11:43:59] INFO Found credentials from IAM Role: credentials.py:1075
Search
ENG IN
18:12 22-11-2024

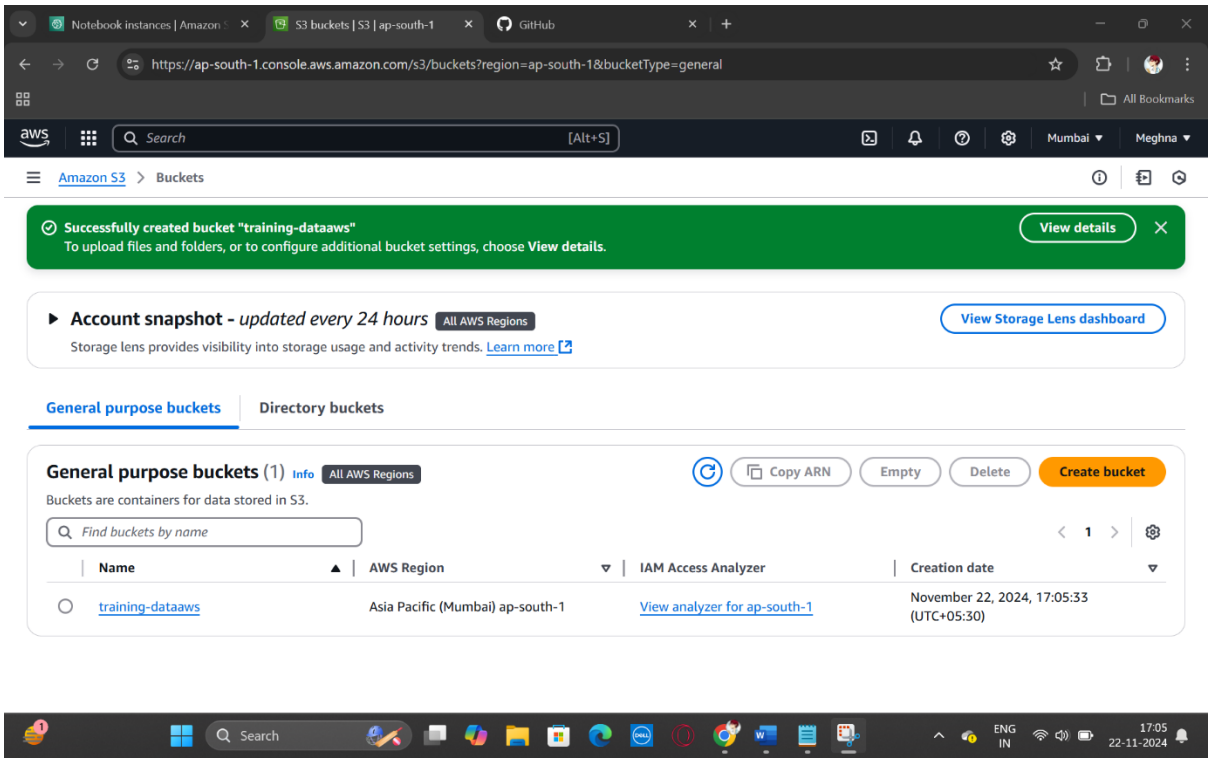

https://aws-project-hraj.notebook.ap-south-1.sagemaker.aws/notebooks/absamodel.ipynb
jupyter absamodel Last Checkpoint: an hour ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
error Trusted
conda_tensorflow2_p310
Run Code nbdiff
[11/22/24 11:43:59] INFO Found credentials from IAM Role: BaseNotebookInstanceEc2InstanceRole credentials.py:1075
[11/22/24 11:44:00] INFO Found credentials from IAM Role: BaseNotebookInstanceEc2InstanceRole credentials.py:1075
done
[11/22/24 11:44:01] INFO Created S3 bucket: sagemaker-ap-south-1-615299742017 session.py:723
INFO Creating model with name: huggingface-pytorch-inference-2024-11-22-11-44-01-938 session.py:4025
[11/22/24 11:44:02] INFO Creating endpoint-config with name huggingface-pytorch-inference-2024-11-22-11-44-02-482 session.py:5820
INFO Creating endpoint with name huggingface-pytorch-inference-2024-11-22-11-44-02-482 session.py:4642
------!
In [3]:
predictor.predict({
"inputs": "The battery life sucks",
})
Out[3]: [{'label': 'negative', 'score': 0.9149718284606934}]
In [ ]:
Search
ENG IN
18:13 22-11-2024

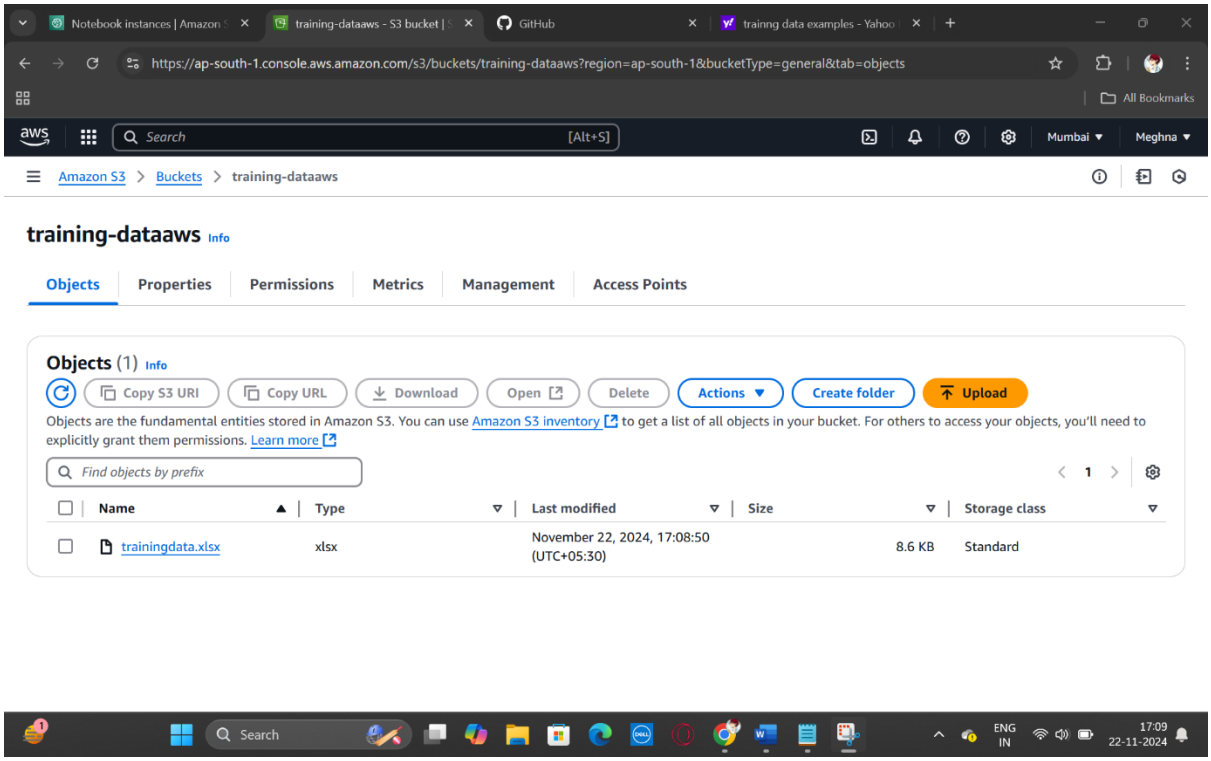

https://ap-south-1.console.aws.amazon.com/sagemaker/home?region=ap-south-1#/endpointConfig
Amazon SageMaker > Endpoint configuration
Endpoint configuration
Apply to endpoint
Clone
Actions
Create endpoint configuration
Search endpoint configuration
Name
ARN
Creation time
huggingface-pytorch-inference-2024-11-22-11-44-02-482
arn:aws:sagemaker:ap-south-1:615299742017:endpoint-config/huggingface-pytorch-inference-2024-11-22-11-44-02-482
11/22/2024, 5:14:02 PM
Endpoint configurations
Endpoints


https://ap-south-1.console.aws.amazon.com/sagemaker/home?region=ap-south-1#/endpoints
Amazon SageMaker > Endpoints
Endpoints
Update endpoint
Actions
Create endpoint
Search endpoints
Name
ARN
Creation time
Status
Last updated
huggingface-pytorch-inference-2024-11-22-11-44-02-482
arn:aws:sagemaker:ap-south-1:615299742017:endpoint/huggingface-pytorch-inference-2024-11-22-11-44-02-482
11/22/2024, 5:14:02 PM
InService
11/22/2024, 5:17:24 PM
Endpoint configurations
Endpoints

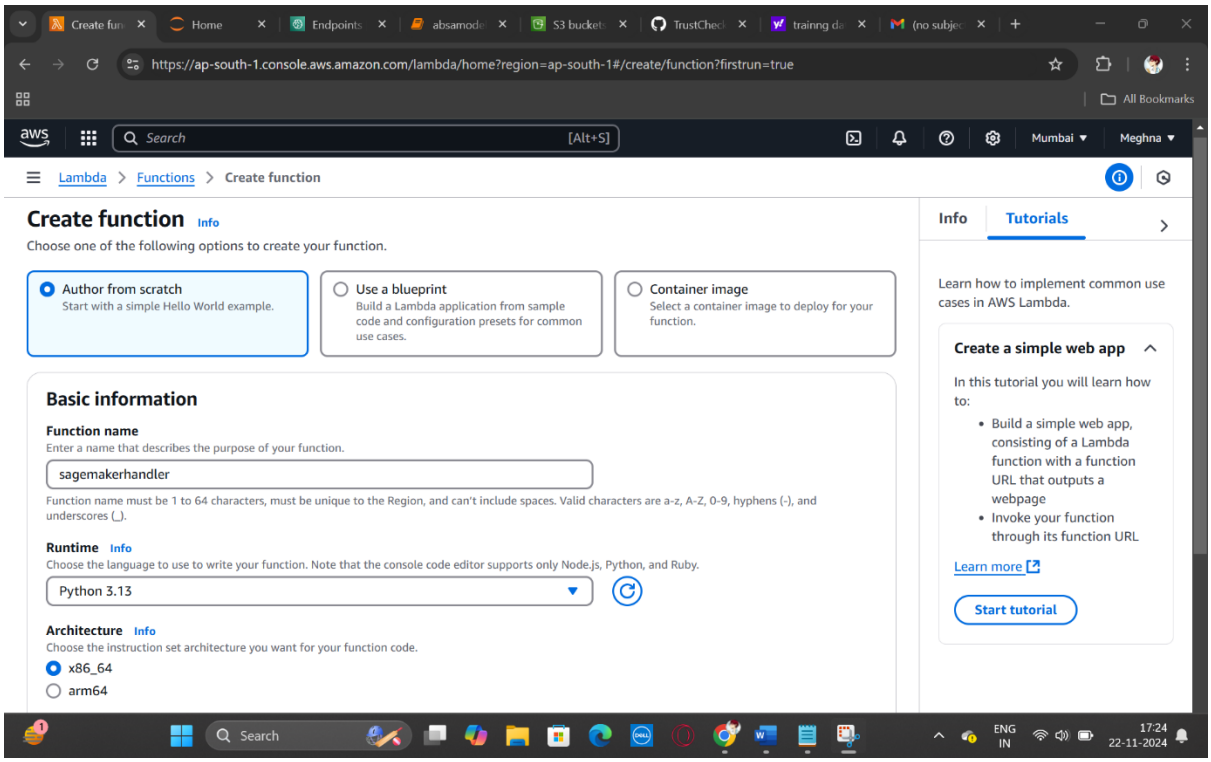
- S3

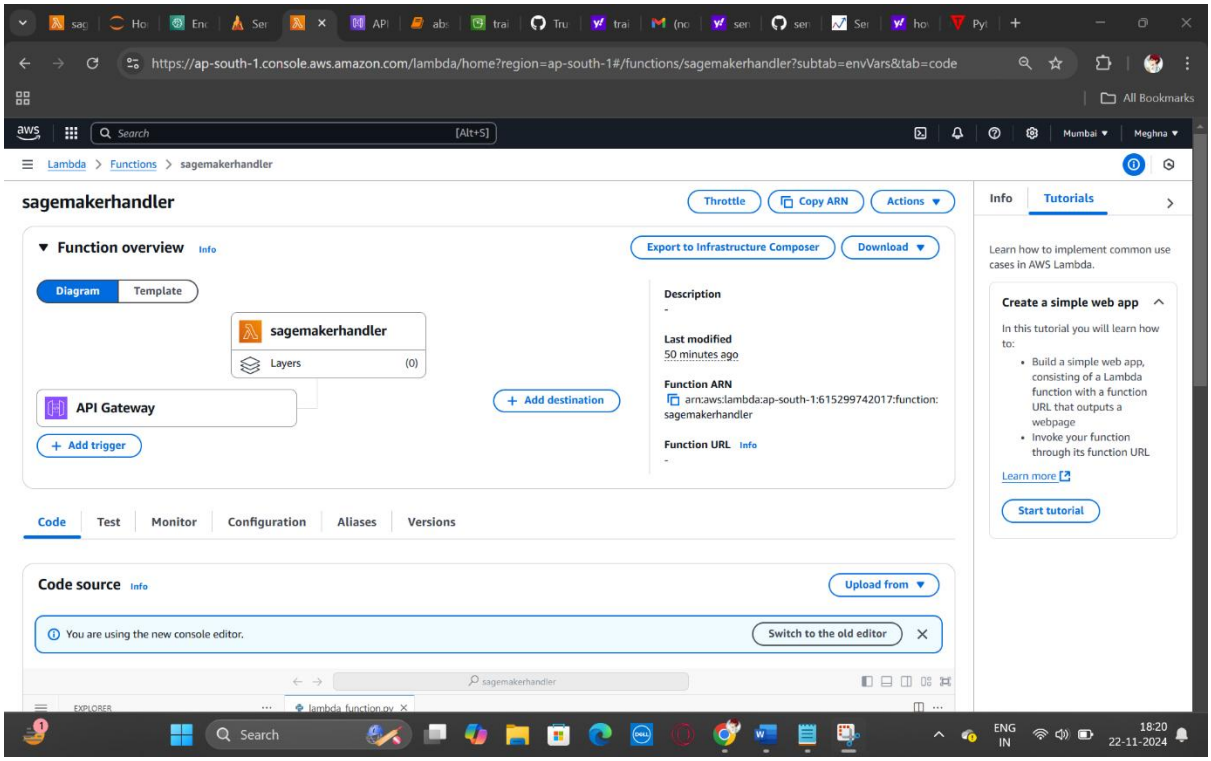

Notebook instances | Amazon
Create S3 bucket | S3 | ap-south
GitHub
https://ap-south-1.console.aws.amazon.com/s3/bucket/create?region=ap-south-1&bucketType=general
All Bookmarks
aws
Search
[Alt+S]
Mumbai
Meghna
Amazon S3 > Buckets > Create bucket
Create bucket Info
Buckets are containers for data stored in S3.
General configuration
AWS Region
Asia Pacific (Mumbai) ap-south-1
Bucket type Info
General purpose
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.
Directory
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.
Bucket name Info
training-dataaws
Bucket name must be unique within the global namespace and follow the bucket naming rules. See rules for bucket naming
Copy settings from existing bucket - optional
Only the bucket settings in the following configuration are copied.
Choose bucket
Format: s3://bucket/prefix
Search
ENG
IN
17:04
22-11-2024

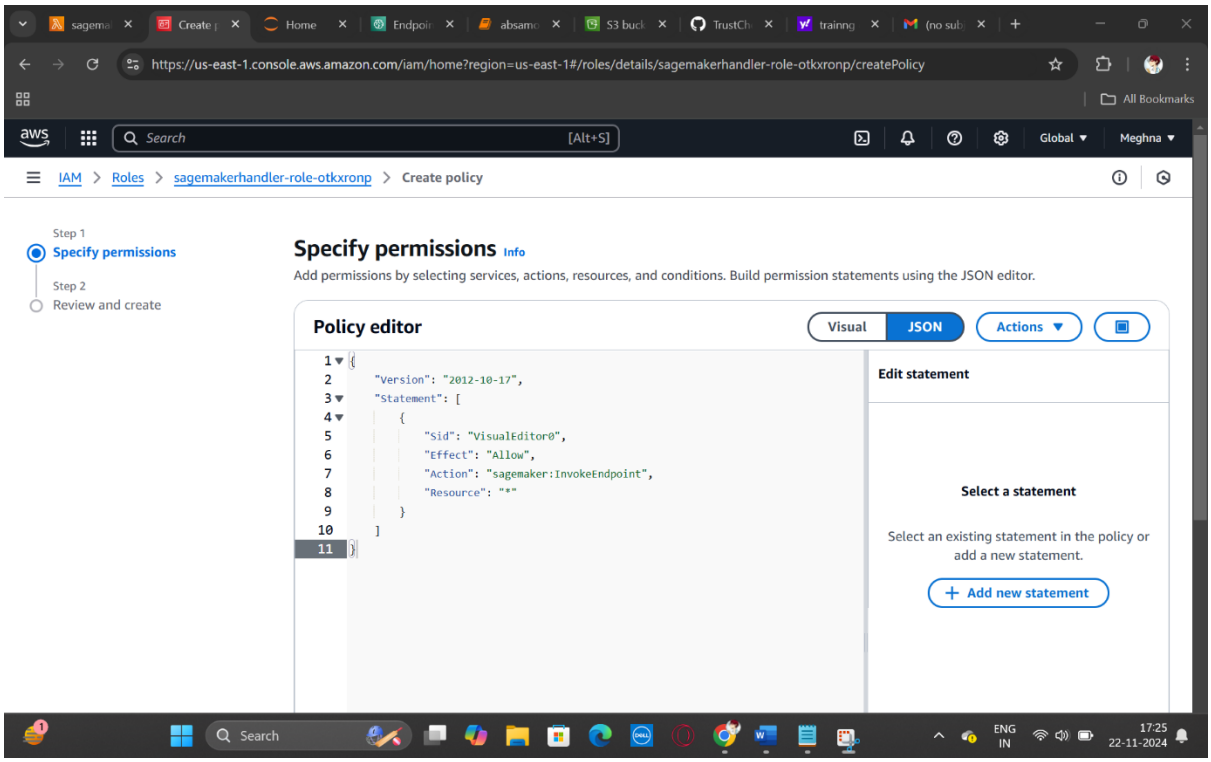

Notebook instances | Amazon
S3 buckets | S3 | ap-south-1
GitHub
https://ap-south-1.console.aws.amazon.com/s3/buckets?region=ap-south-1&bucketType=general
All Bookmarks
aws
Search
[Alt+S]
Mumbai
Meghna
Amazon S3 > Buckets
Successfully created bucket "training-dataaws"
To upload files and folders, or to configure additional bucket settings, choose View details.
View details
Account snapshot - updated every 24 hours
All AWS Regions
Storage lens provides visibility into storage usage and activity trends. Learn more
View Storage Lens dashboard
General purpose buckets
Directory buckets
General purpose buckets (1) Info
All AWS Regions
Buckets are containers for data stored in S3.
Copy ARN
Empty
Delete
Create bucket
Find buckets by name
1
Name
AWS Region
IAM Access Analyzer
Creation date
training-dataaws
Asia Pacific (Mumbai) ap-south-1
View analyzer for ap-south-1
November 22, 2024, 17:05:33 (UTC+05:30)
Search
ENG
IN
17:05
22-11-2024

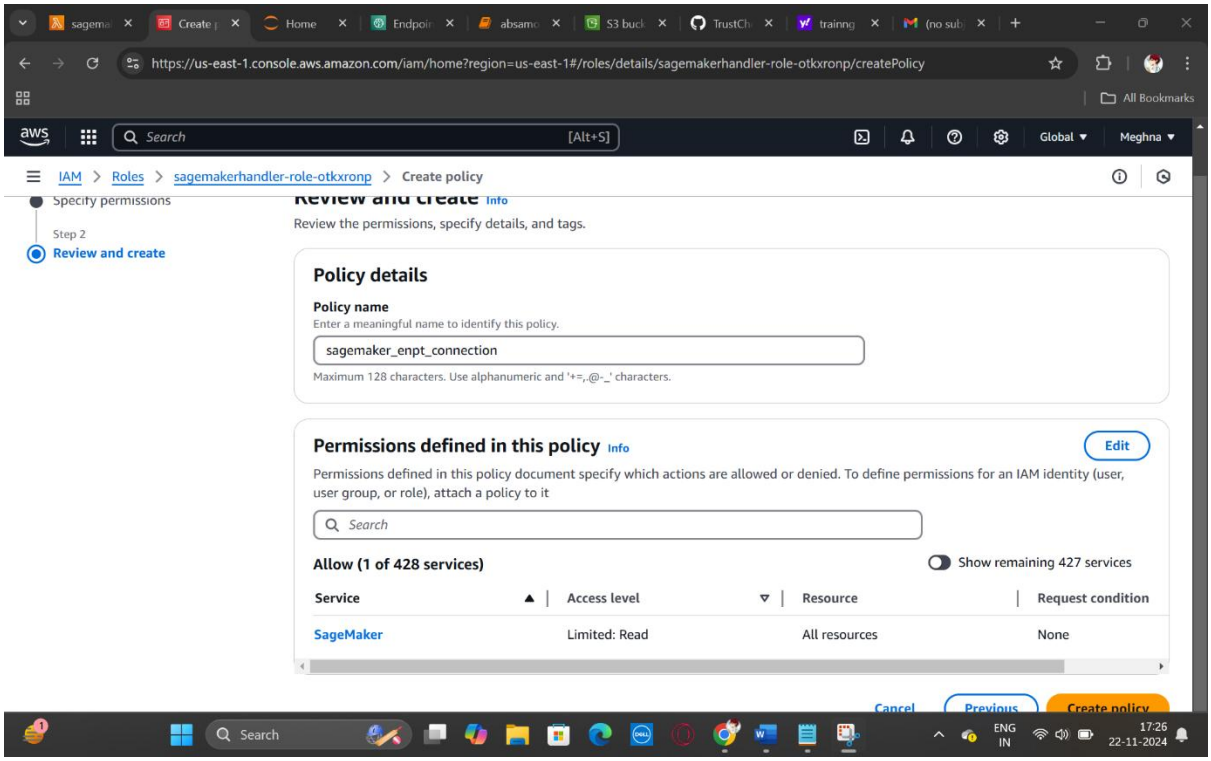

Notebook instances | Amazon
training-dataaws - S3 bucket
GitHub
trainng data examples - Yahoo
https://ap-south-1.console.aws.amazon.com/s3/buckets/training-dataaws?region=ap-south-1&bucketType=general&tab=objects
All Bookmarks
aws
Search
[Alt+S]
Mumbai
Meghna
Amazon S3 > Buckets > training-dataaws
training-dataaws Info
Objects
Properties
Permissions
Metrics
Management
Access Points
Objects (1) Info
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Find objects by prefix
Name
Type
Last modified
Size
Storage class
trainingdata.xlsx
xlsx
November 22, 2024, 17:08:50 (UTC+05:30)
8.6 KB
Standard
Search
ENG IN
17:09
22-11-2024

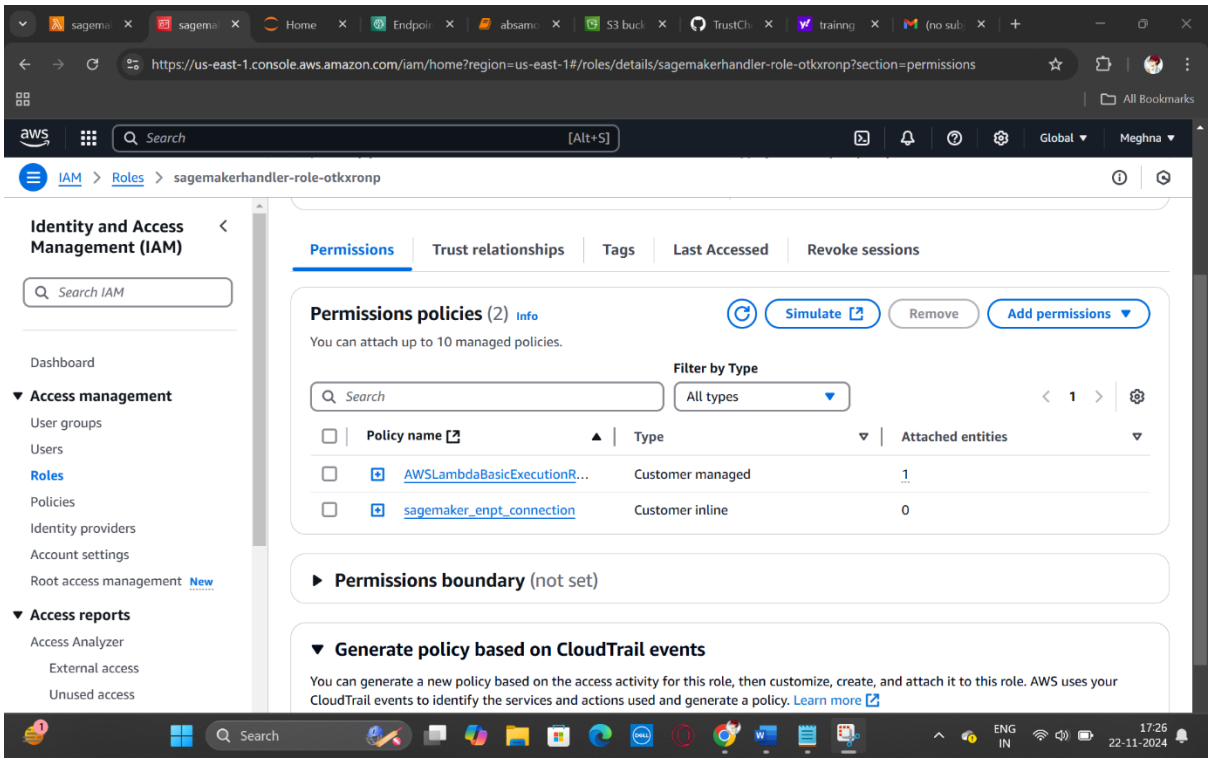
- Lambda

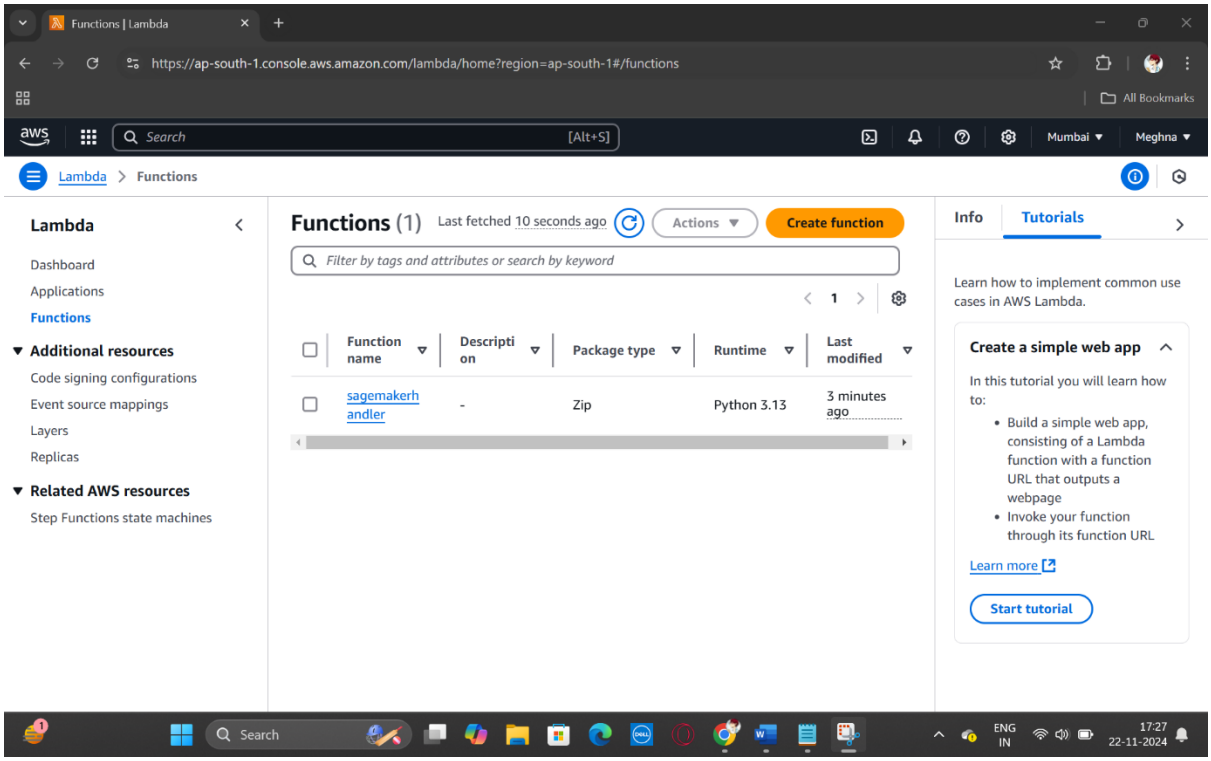

Create fun
Home
Endpoints
absamode
S3 buckets
TrustChec
training da
(no subjec
https://ap-south-1.console.aws.amazon.com/lambda/home?region=ap-south-1#/create/function?firstrun=true
All Bookmarks
aws
Search
[Alt+S]
Mumbai
Meghna
Lambda > Functions > Create function
Create function Info
Choose one of the following options to create your function.
Author from scratch
Start with a simple Hello World example.
Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.
Container image
Select a container image to deploy for your function.
Basic information
Function name
Enter a name that describes the purpose of your function.
sagemakerhandler
Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (_).
Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.13
Architecture Info
Choose the instruction set architecture you want for your function code.
x86_64
arm64
Info
Tutorials
Learn how to implement common use cases in AWS Lambda.
Create a simple web app
In this tutorial you will learn how to:
Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
Invoke your function through its function URL
Learn more
Start tutorial
Search
ENG IN
17:24
22-11-2024

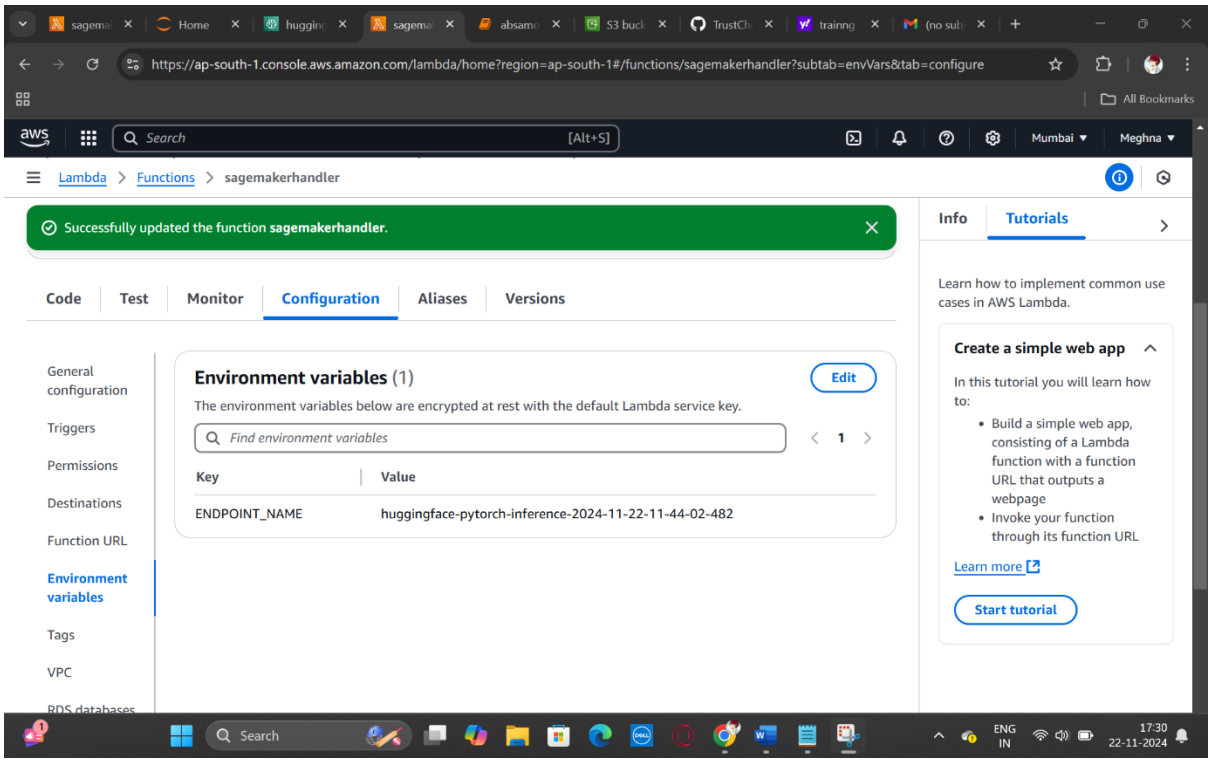

https://ap-south-1.console.aws.amazon.com/lambda/home?region=ap-south-1#/functions/sagemakerhandler?subtab=envVars&tab=code
Lambda > Functions > sagemakerhandler
sagemakerhandler
Throttle
Copy ARN
Actions
Function overview
Info
Export to Infrastructure Composer
Download
Diagram
Template
sagemakerhandler
Layers
(0)
API Gateway
Add trigger
Add destination
Description
-
Last modified
50 minutes ago
Function ARN
arn:aws:lambda:ap-south-1:615299742017:function:sagemakerhandler
Function URL
Info
-
Code
Test
Monitor
Configuration
Aliases
Versions
Code source
Info
Upload from
You are using the new console editor.
Switch to the old editor
Info
Tutorials
Learn how to implement common use cases in AWS Lambda.
Create a simple web app
In this tutorial you will learn how to:
Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
Invoke your function through its function URL
Learn more
Start tutorial

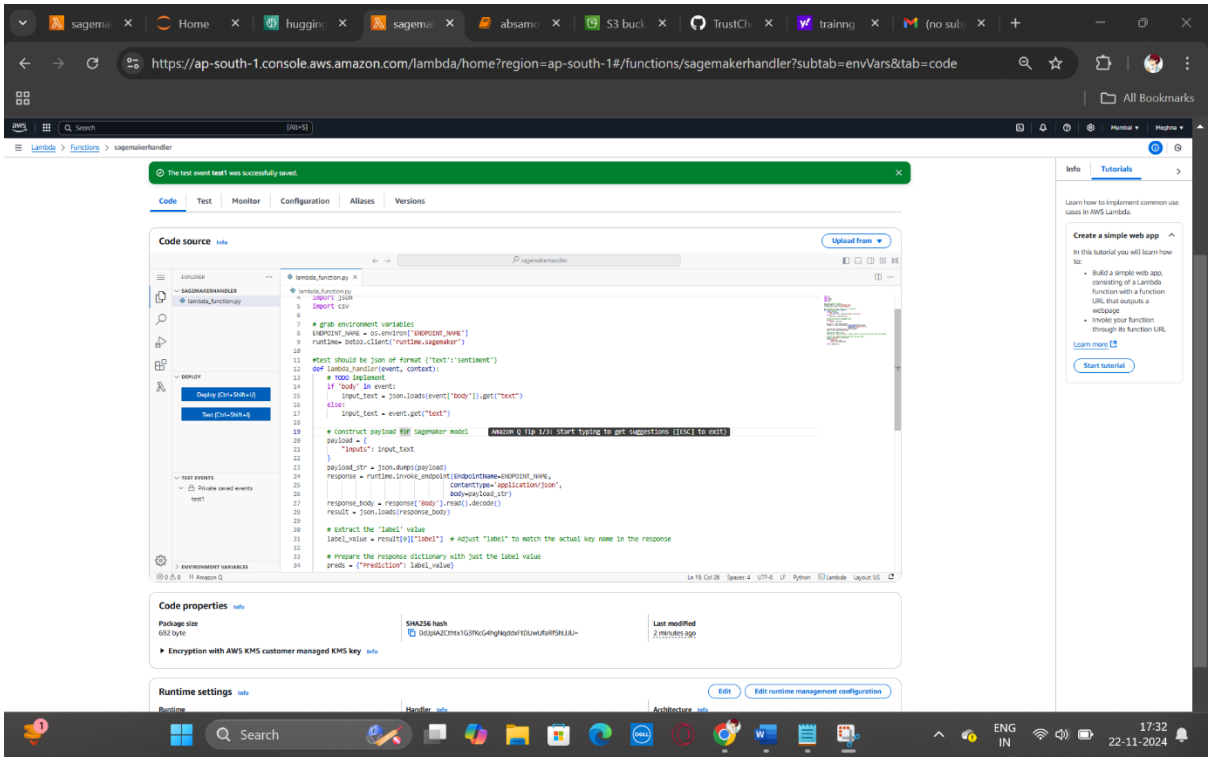

https://us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/roles/details/sagemakerhandler-role-otkxronp/createPolicy
IAM > Roles > sagemakerhandler-role-otkxronp > Create policy
Step 1
Specify permissions
Step 2
Review and create
Specify permissions
Info
Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.
Policy editor
Visual
JSON
Actions
{
"Version": "2012-10-17",
"Statement": [
{
"Sid": "VisualEditor0",
"Effect": "Allow",
"Action": "sagemaker:InvokeEndpoint",
"Resource": "*"
}
]
}
Edit statement
Select a statement
Select an existing statement in the policy or add a new statement.
Add new statement


https://us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/roles/details/sagemakerhandler-role-otkxronp/createPolicy
IAM > Roles > sagemakerhandler-role-otkxronp > Create policy
Specify permissions
Step 2
Review and create
Review the permissions, specify details, and tags.
Policy details
Policy name
Enter a meaningful name to identify this policy.
sagemaker_enpt_connection
Maximum 128 characters. Use alphanumeric and '+=,.@-_' characters.
Permissions defined in this policy Info
Edit
Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it
Search
Allow (1 of 428 services)
Show remaining 427 services
Service
Access level
Resource
Request condition
SageMaker
Limited: Read
All resources
None
Cancel
Previous
Create policy

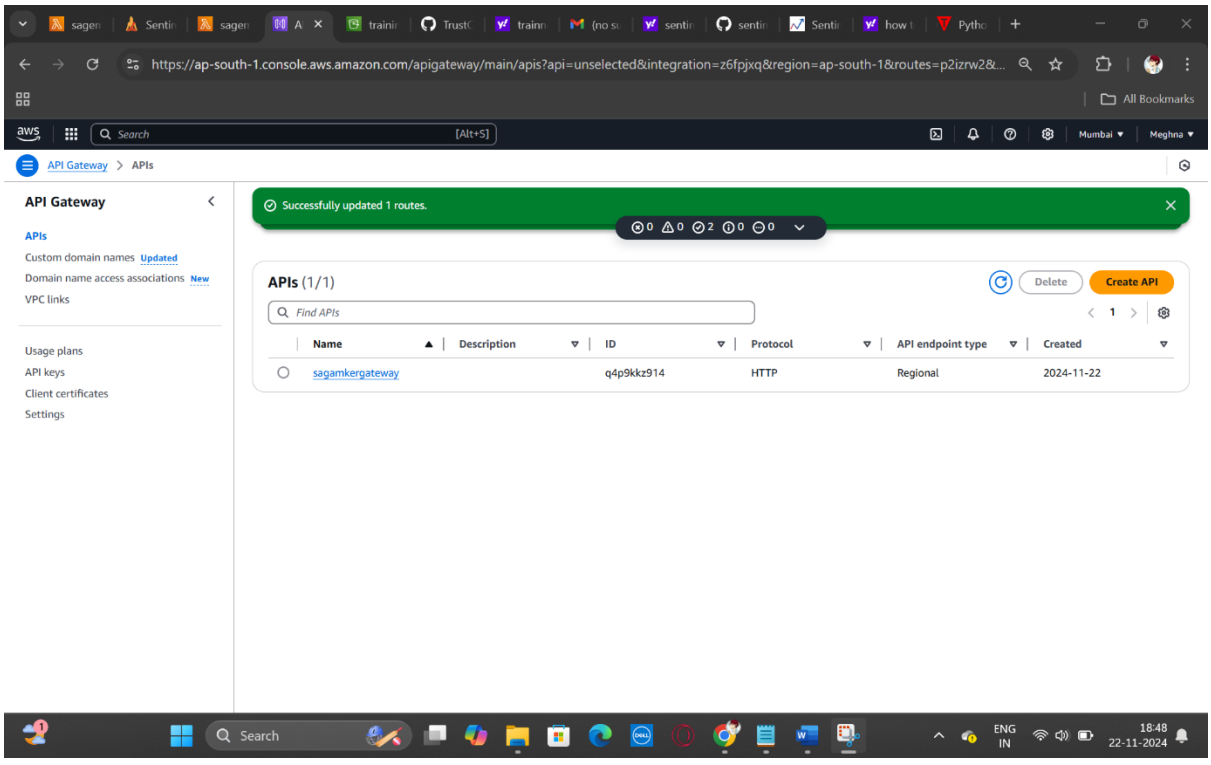

https://us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/roles/details/sagemakerhandler-role-otkxronp?section=permissions
IAM > Roles > sagemakerhandler-role-otkxronp
Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Root access management New
Access reports
Access Analyzer
External access
Unused access
Permissions
Trust relationships
Tags
Last Accessed
Revoke sessions
Permissions policies (2) Info
You can attach up to 10 managed policies.
Simulate
Remove
Add permissions
Filter by Type
Search
All types
Policy name
Type
Attached entities
AWSLambdaBasicExecutionR...
Customer managed
1
sagemaker_enpt_connection
Customer inline
0
Permissions boundary (not set)
Generate policy based on CloudTrail events
You can generate a new policy based on the access activity for this role, then customize, create, and attach it to this role. AWS uses your CloudTrail events to identify the services and actions used and generate a policy. Learn more

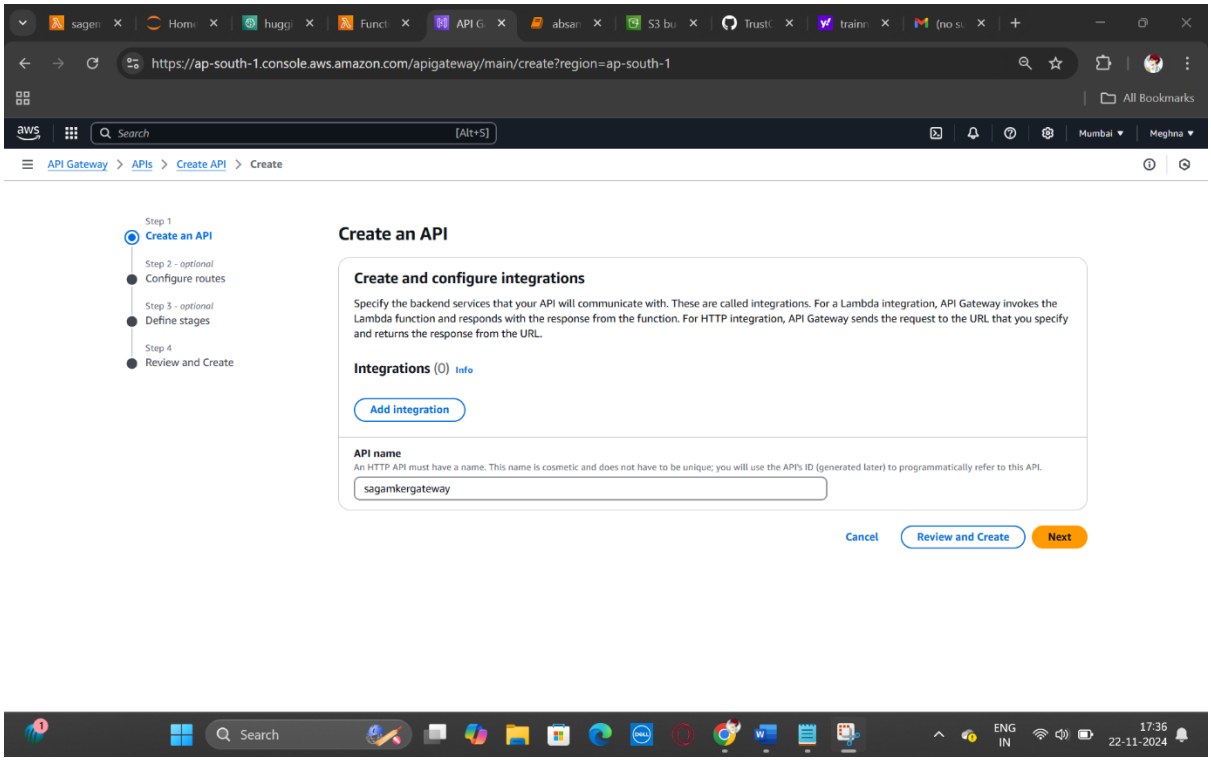

Functions | Lambda
https://ap-south-1.console.aws.amazon.com/lambda/home?region=ap-south-1#/functions
Search [Alt+S]
Mumbai
Meghna
Lambda > Functions
Lambda
Dashboard
Applications
Functions
Additional resources
Code signing configurations
Event source mappings
Layers
Replicas
Related AWS resources
Step Functions state machines
Functions (1)
Last fetched 10 seconds ago
Actions
Create function
Filter by tags and attributes or search by keyword
Function name
Description
Package type
Runtime
Last modified
sagemakerhandler
-
Zip
Python 3.13
3 minutes ago
Info
Tutorials
Learn how to implement common use cases in AWS Lambda.
Create a simple web app
In this tutorial you will learn how to:
Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
Invoke your function through its function URL
Learn more
Start tutorial
Search
ENG IN
17:27
22-11-2024


https://ap-south-1.console.aws.amazon.com/lambda/home?region=ap-south-1#/functions/sagemakerhandler?subtab=envVars&tab=configure
Search [Alt+S]
Mumbai
Meghna
Lambda > Functions > sagemakerhandler
Successfully updated the function sagemakerhandler.
Code
Test
Monitor
Configuration
Aliases
Versions
General configuration
Triggers
Permissions
Destinations
Function URL
Environment variables
Tags
VPC
RDS databases
Environment variables (1)
Edit
The environment variables below are encrypted at rest with the default Lambda service key.
Find environment variables
Key
Value
ENDPOINT_NAME
huggingface-pytorch-inference-2024-11-22-11-44-02-482
Info
Tutorials
Learn how to implement common use cases in AWS Lambda.
Create a simple web app
In this tutorial you will learn how to:
Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
Invoke your function through its function URL
Learn more
Start tutorial
Search
ENG IN
17:30
22-11-2024

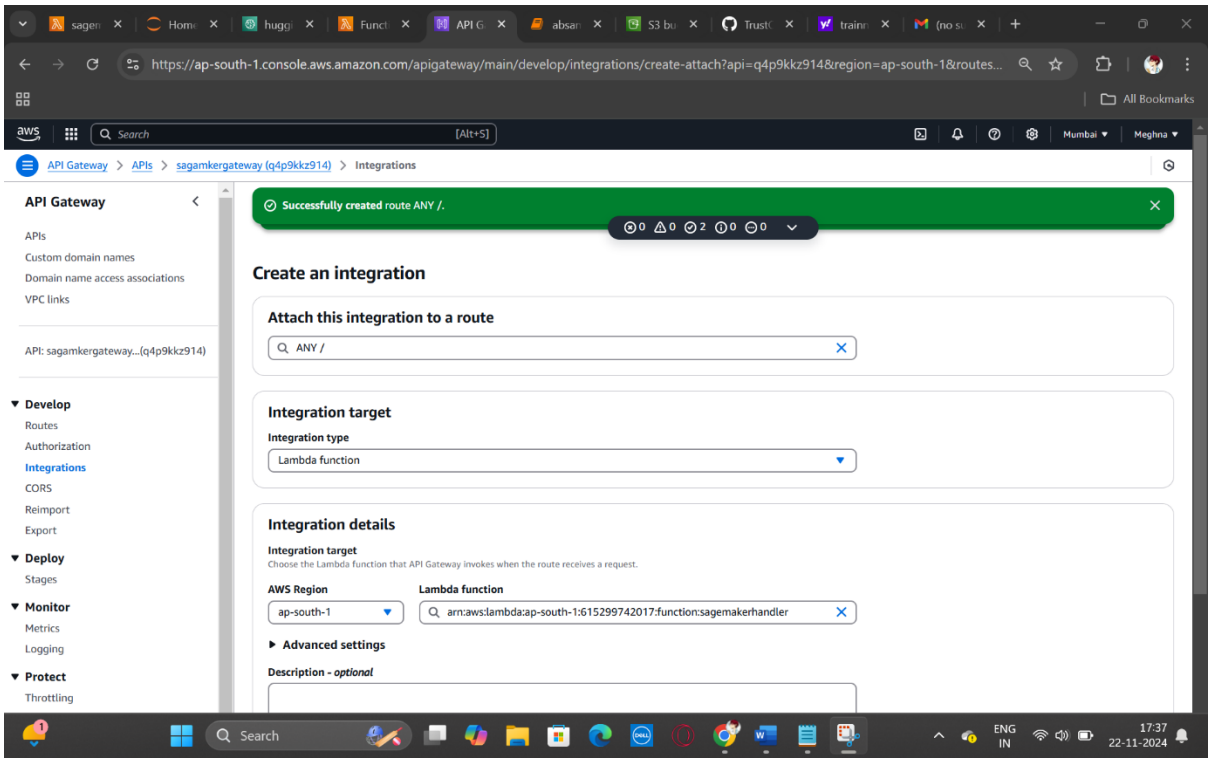

https://ap-south-1.console.aws.amazon.com/lambda/home?region=ap-south-1#/functions/sagemakerhandler?subtab=envVars&tab=code
Code source
Code properties
Runtime settings

- Api gateway

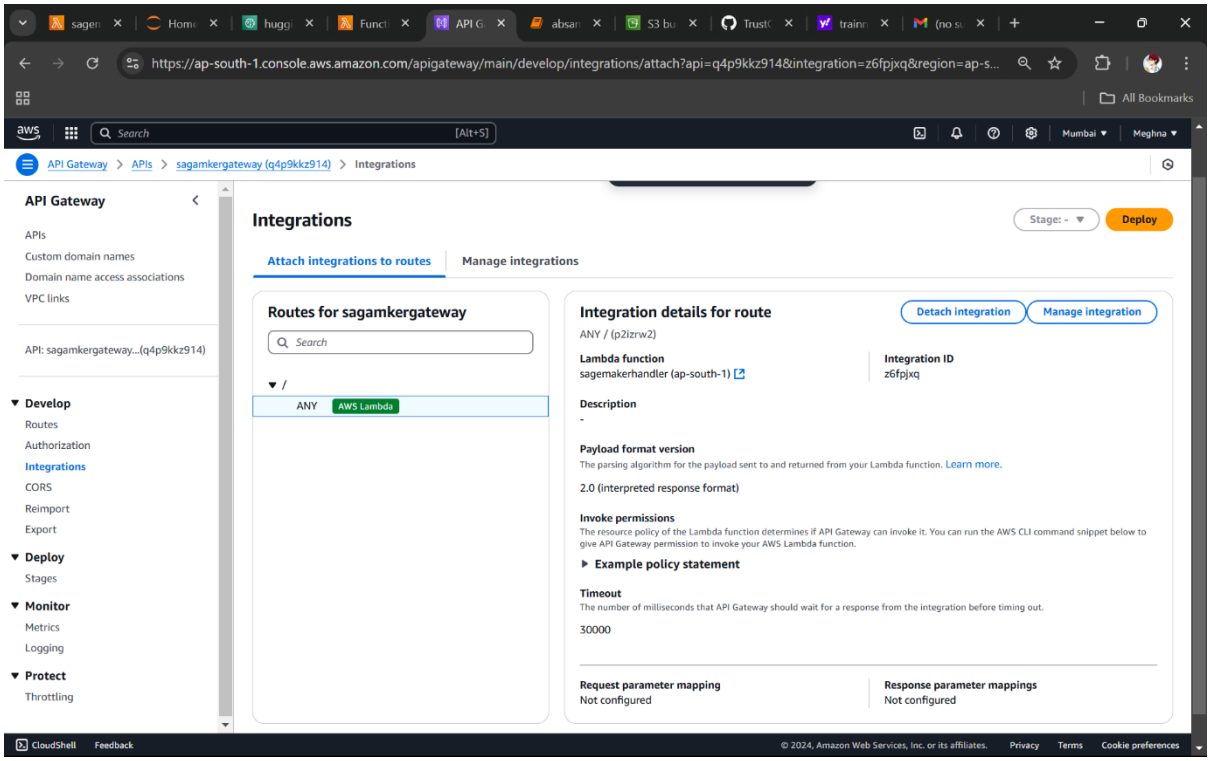

https://ap-south-1.console.aws.amazon.com/apigateway/main/apis?api=unselected&integration=z6fpjxq&region=ap-south-1&routes=p2izrw2&...
API Gateway
Successfully updated 1 routes.
APIs (1/1)
sagamkergateway
q4p9kkz914
HTTP
Regional
2024-11-22

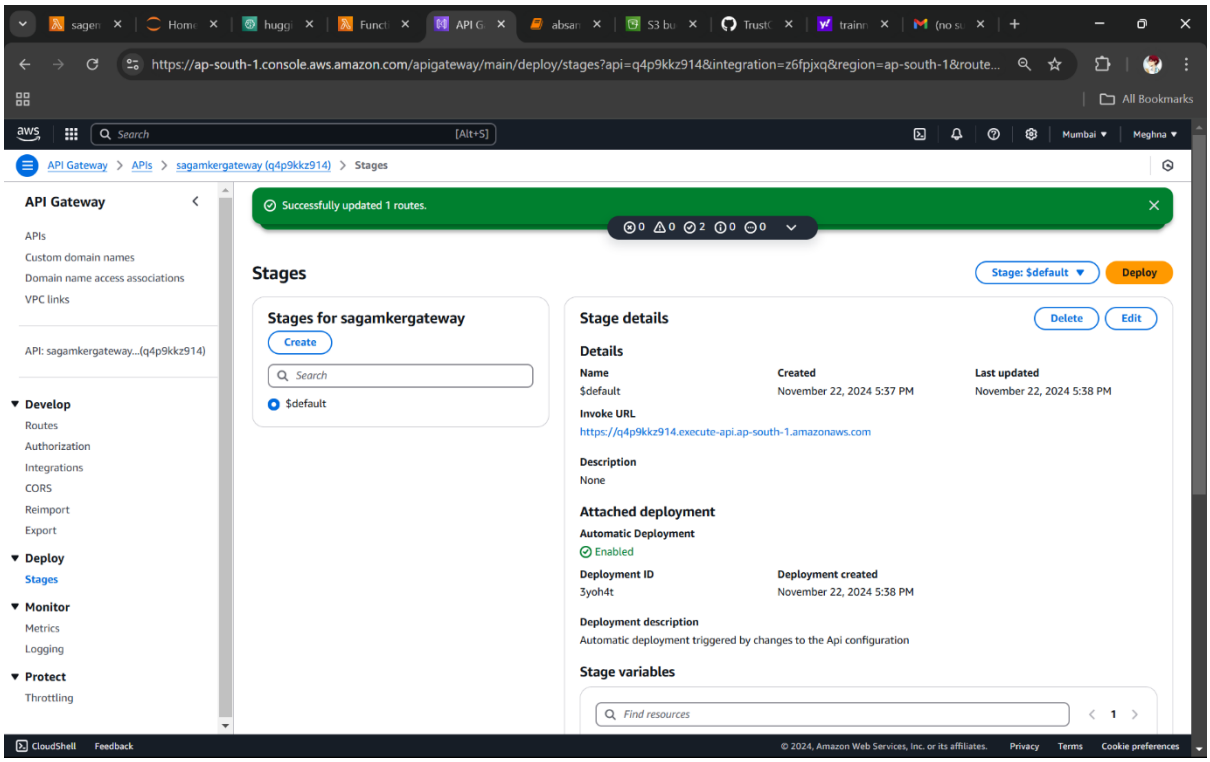

https://ap-south-1.console.aws.amazon.com/apigateway/main/create?region=ap-south-1
API Gateway > APIs > Create API > Create
Step 1
Create an API
Step 2 - optional
Configure routes
Step 3 - optional
Define stages
Step 4
Review and Create
Create an API
Create and configure integrations
Specify the backend services that your API will communicate with. These are called integrations. For a Lambda integration, API Gateway invokes the Lambda function and responds with the response from the function. For HTTP integration, API Gateway sends the request to the URL that you specify and returns the response from the URL.
Integrations (0) Info
Add integration
API name
An HTTP API must have a name. This name is cosmetic and does not have to be unique; you will use the API's ID (generated later) to programmatically refer to this API.
sagamkergateway
Cancel
Review and Create
Next

Search
ENG
IN
17:36
22-11-2024


https://ap-south-1.console.aws.amazon.com/apigateway/main/develop/integrations/create-attach?api=q4p9kkz914&region=ap-south-1&routes...
API Gateway > APIs > sagamkergateway (q4p9kkz914) > Integrations
API Gateway
APIs
Custom domain names
Domain name access associations
VPC links
API: sagamkergateway...(q4p9kkz914)
Develop
Routes
Authorization
Integrations
CORS
Reimport
Export
Deploy
Stages
Monitor
Metrics
Logging
Protect
Throttling
Successfully created route ANY /.
Create an integration
Attach this integration to a route
ANY /
Integration target
Integration type
Lambda function
Integration details
Integration target
Choose the Lambda function that API Gateway invokes when the route receives a request.
AWS Region
ap-south-1
Lambda function
arn:aws:lambda:ap-south-1:615299742017:function:sagemakerhandler
Advanced settings
Description - optional

Search
ENG
IN
17:37
22-11-2024





| Project Code

I. Browser extension:

1. manifest.json

```
{
  "manifest_version": 3,
  "name": "Amazon Reviews Fetcher",
  "version": "1.0",
  "permissions": ["activeTab", "scripting","alarms"],
  "action": {
    "default_popup": "popup.html"
  },
  "content_scripts": [
    {
      "matches": ["<all_urls>"],
      "js": ["content.js"]
    }
  ],
  "icons": {
    "128": "128.png"
  }
}
```

2. popup.js

```
document.getElementById('fetch-reviews').addEventListener('click', () => {
  var aspect = document.getElementById("messageInput").value;

  let mes = ""
  if(aspect == ""){ mes = "alertMessage" }
  else{ mes = aspect}

  chrome.tabs.query({ active: true, currentWindow: true }, (tabs) => {
     chrome.tabs.sendMessage(tabs[0].id, { message: mes });
  });
```

```
    chrome.runtime.onMessage.addListener((message, sender, sendResponse) => {
        if (message.verdict) {
            console.log("New message received: ", message.verdict);
            document.getElementById("out").innerHTML = message.verdict;
        }
    });
});
```

3. popup.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Amazon Review Fetcher</title>
  <style>
    body{
        display: flex;
        width: 300px;
        background-color: black;
        flex-direction: column;
        font-family: monospace;
        color: aliceblue;
    }
    h1{
        margin-top: 0;
        text-align: center;
        font-size: 2.5em;
        border-bottom: 1px solid white;
        padding-top: 10px;
        padding-bottom: 12px;
    }
    #asp{
        font-size: 2em;
    }
```

```
#fetch-reviews{
    padding: 10px;
    margin:auto;
    width: 50%;
    font-family: monospace;
    background-color: black;
    color: aliceblue;
    box-shadow: 5px 5px aliceblue;
}
#fetch-reviews:active {
    box-shadow: 0 0 #666;
    transform: translateY(5px);
    transform: translateX(5px);
}
#butdiv{
    margin-top: 16px;
    display: flex;
}
input{
    width: 200px;
    height: 30px;
    background-color: black;
    border: 1px white solid;
    color: white;
    border-radius: 2px;
    font-family: monospace;
}
#verdict{
    background-color: aliceblue;
    color: black;
    margin-top: 14px;
    padding: 12px;
```

```
        padding-left: 5px;
        padding-right: 5px;
        line-height: 1.5;
    }
    #ver{
        background-color: aliceblue;
        font-size: 1.8em;
        font-weight: bold;
    }
    #out{
        font-size: 1.5em;
    }
  </style>
</head>
<body>
  <h1>TRUSTCHECK</h1>
  <div>
    <label for="messageInput" id = "asp">Aspect:  </label>
    <input type="text" id="messageInput" placeholder="Enter your Aspect" />
  </div>
  <div id = "butdiv">
    <button id="fetch-reviews">Fetch Reviews</button>
  </div>
  <div id = "verdict">
    <span id = "ver">Vedict: </span>
    <span id = "out"></span>
  </div>
  <script src="popup.js"></script>
</body>
</html>
```

4. content.js

```
import {
```

```
    SecretsManagerClient,
    GetSecretValueCommand,
  } from "@aws-sdk/client-secrets-manager";

async function getReview(){
    const reviews = [];

    while (true) {
        document.querySelectorAll('.review').forEach(reviewElement => {
            const reviewText = reviewElement.querySelector('.review-text').innerText;
            reviews.push(reviewText);
        });
        const nextButtonContainer = document.querySelector('li.a-last');
        if (nextButtonContainer && nextButtonContainer.classList.contains('a-disabled')) {
            break;
        } else if (nextButtonContainer) {
            nextButtonContainer.querySelector('a').click();
            console.log("click")
            await new Promise(resolve => setTimeout(resolve, 2000));
        } else {
            console.log("Next button not found.");
            break;
        }
    }
    console.log("All Reviews Collected:", reviews);
    return reviews
}


chrome.runtime.onMessage.addListener(async (request, sender, sendResponse) => {
    if(request.message === "alertMessage") {
        alert("Enter a aspect")
```

```
}
else{
    let arr = await getReview()
    const myHeaders = new Headers();
    myHeaders.append("Content-Type", "application/json");

    const raw = JSON.stringify({
     sentance: arr,
     aspect: request.message
    });

    const requestOptions = {
    method: "POST",
    headers: myHeaders,
    body: raw,
    redirect: "follow",
    };

    const secret_name = "SECRET_KEY";

    const client = new SecretsManagerClient({
    region: "ap-south-1",
    });

    let response;

    try {
    response = await client.send(
        new GetSecretValueCommand({
        SecretId: secret_name,
        VersionStage: "AWSCURRENT",
        })
```

```
        );
        } catch (error) { throw error;}

        const secret = response.SecretString;

        fetch(secret, requestOptions)
        .then((response) => response.text())
        .then((result) => {
            console.log(result)
            chrome.runtime.sendMessage({ verdict: JSON.parse(result).verdict });
        })
        .catch((error) => {
            console.error(error)
            chrome.runtime.sendMessage({ verdict: "failure" });
        });
    }
});
```

II. Node JS Server

1. App.js

```
const http = require('node:http');
const express = require('express');
const cors = require('cors');
var emojiStrip = require('emoji-strip');
const LanguageDetect = require('languagedetect');
const langDetect = new LanguageDetect();
langDetect.setLanguageType('iso2');

const port = 80;

const app = express();
app.use(cors());
```

```
app.use(express.json());

app.post('/', (req, res) => {
    let sen = req.body.sentance
    let asp = req.body.aspect

    sen = sen.map(sentence => emojiStrip(sentence.replace(/\n/g, ' ')));
    console.log(sen);

    const myHeaders = new Headers();
    myHeaders.append("Content-Type", "application/json");

    const raw = JSON.stringify({
        text: sen[0]
    });

    const requestOptions = {
        method: "POST",
        headers: myHeaders,
        body: raw,
        redirect: "follow",
    };

    let result = {}

    fetch("https://q4p9kkz914.execute-api.ap-south-1.amazonaws.com", requestOptions)
    .then((response) => response.text())
    .then((result) => {
        console.log(result)
        final = JSON.parse(result)
        result = {
            verdict: final.Prediction,
```

```
        };

        res.json(result);
    })
    .catch((error) => {
        console.error(error)
        final = "error"
        result = {
            verdict: final,
        };

        res.json(result);
    });
});

app.listen(port, () => {
    console.log(`Server running on port ${port}`);
});
```

III. Sagemaker

```
!pip install transformers
!pip install -U sagemaker
import tensorflow as tf
from tensorflow.keras.preprocessing import sequence
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, SimpleRNN, Dense, Flatten
from sklearn.preprocessing import LabelEncoder
import pandas as pd
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import to_categorical

def trainmodel(path):
```

```
db = pd.read_excel(path,encoding='latin1')
db = db.drop(['textID'],axis = 1)
y = db['sentiment']
X = db['selected_text']
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.2,random_state=42)
db.head()
X_train = X_train.astype(str)
X_test = X_test.astype(str)
tokenizer = tf.keras.preprocessing.text.Tokenizer(num_words=10000)
tokenizer.fit_on_texts(pd.concat([X_train, X_test]))
X_train_seq = tokenizer.texts_to_sequences(X_train)
X_test_seq = tokenizer.texts_to_sequences(X_test)

max_length = 500
X_train_pad = sequence.pad_sequences(X_train_seq, maxlen=max_length)
X_test_pad = sequence.pad_sequences(X_test_seq, maxlen=max_length)

label_encoder = LabelEncoder()
y_train_encoded = to_categorical(label_encoder.fit_transform(y_train))
y_test_encoded = to_categorical(label_encoder.transform(y_test))

model = Sequential()
model.add(Embedding(10000, 32, input_length=max_length))
model.add(SimpleRNN(32))
model.add(Flatten()) #additional flatten
model.add(Dense(64, activation='relu')) #additional dense
model.add(Dense(3, activation='softmax'))

model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(X_train_pad, y_train_encoded, epochs=10, batch_size=128, validation_split=0.2)
return model
```

```
s3_path=’s3://training-dataaws.trainingdata.xlsv’
model=trainmodel(path)

import sagemaker
import boto3
try:
	role = sagemaker.get_execution_role()
except ValueError:
	iam = boto3.client('iam')
	role = iam.get_role(RoleName='sagemaker_execution_role')['Role']['Arn']

hub = {
	'HF_MODEL_ID':'cardiffnlp/twitter-roberta-base-sentiment-latest',
	'HF_TASK':'text-classification'
}

# create Hugging Face Model Class
deploy_model = model(
	transformers_version='4.37.0',
	pytorch_version='2.1.0',
	py_version='py310',
	env=hub,
	role=role,
)

print("done")

# deploy model to SageMaker Inference
predictor =deploy_model.deploy(
	initial_instance_count=1, # number of instances
	instance_type='ml.m5.xlarge' # ec2 instance type
)
```

```
predictor.predict({
        "inputs": "The battery life sucks",
})
```

IV. Lambda

```
    result = json.loads(response_body)

    # Extract the 'label' value
    label_value = result[0]["label"]  # Adjust "label" to match the actual key name in the response

    # Prepare the response dictionary with just the label value
    preds = {"Prediction": label_value}
    response_dict = {
        "statusCode": 200,
        "body": json.dumps(preds)
            }
    return response_dict
#permission
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "sagemaker:InvokeEndpoint",
            "Resource": "*"
        }
    ]
}
```

| References

https://docs.aws.amazon.com/ec2/

https://docs.aws.amazon.com/lambda/

https://docs.aws.amazon.com/s3/

https://docs.aws.amazon.com/sagemaker/

https://docs.aws.amazon.com/apigateway/

https://docs.aws.amazon.com/secretsmanager/

https://docs.aws.amazon.com/iam/

https://paperswithcode.com/paper/effective-lstms-for-target-dependent

https://paperswithcode.com/paper/attention-based-lstm-for-aspect-level