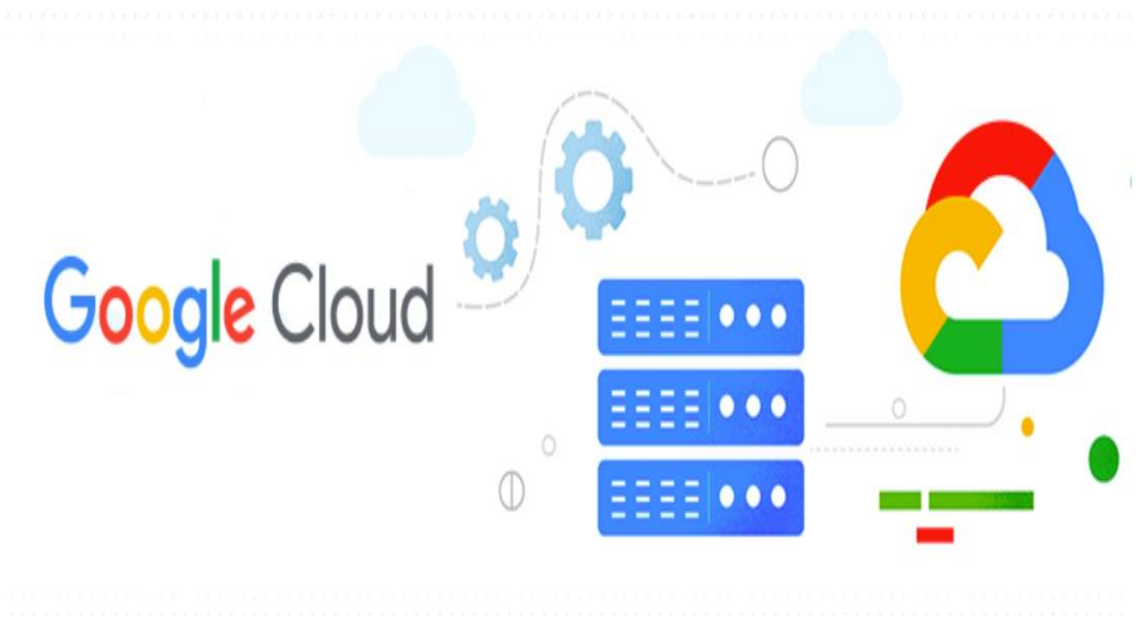
CREATE A CLOUD FUNCTION TO PULL A MESSAGE FROM PUB/SUB

A Project Report Submitted in the fulfilment of the requirements for

Final-Project Evaluation



Name: Sanjeev M

Employee id: 2320436

GenC Intern

CONTENTS

1.OBJECTIVES

- The objective of this project is to build a real-time stock price monitoring system that alerts users when a stock price crosses a certain threshold.
- The system will use Google Cloud Pub/Sub to stream real-time stock price updates and Google Cloud Functions to analyse and send alerts.
- By accomplishing these objectives, this project aims to empower organizations to leverage GCP's storage capabilities effectively, driving operational efficiency, and enhancing data governance practices.

2.INTRODUCTION

In the ever-expanding landscape of cloud computing, efficient data storage management is paramount for organizations to balance performance, accessibility, and cost-effectiveness. Google Cloud Platform (GCP) offers a comprehensive suite of storage solutions tailored to meet diverse needs, including Standard and Archive buckets, each optimized for specific use cases.

Google Cloud Platform (GCP) offered by Google, is a suite of cloud computing services that Provides a series of modular cloud services including computing, data storage, data analytics, and machine learning, artificial intelligence alongside a set of management tools.

GCP run on the same infrastructure that Google uses internally for its own products, such as Google Search and YouTube.
GCP is designed to be scalable, reliable, and secure. It is also cost-effective, with a pay-as-you-go pricing model.

Here are some key points about GCP:

1. **GCP Services**: GCP offers a wide range of infrastructure and application services that can be accessed on-demand. It enables users to build, deploy, and scale applications seamlessly while taking advantage of Google's powerful and reliable infrastructure.
2. **Global Resources**: GCP consists of physical assets (such as computers and hard disk drives) and virtual resources (such as virtual machines) distributed across data centres worldwide. These resources are organized into regions and zones, providing redundancy and reduced latency for better performance.

Services and Integration: When you develop applications on GCP, you combine various services to create the infrastructure you need. GCP services include compute, storage, databases, machine learning, and more. Additionally, GCP integrates seamlessly with other Google Cloud products.

Benefits of using GCP:

There are many benefits to using GCP, including:

- **Scalability:** GCP can scale up or down to meet the demands of your application.

- **Reliability:** GCP is designed to be highly reliable, with a 99.9% uptime guarantee.
- **Security:** GCP is one of the most secure cloud platforms available, with a variety of security features to protect your data.
- **Cost-effectiveness:** GCP is a cost-effective cloud platform, with a pay-as-you-go pricing model.

Getting started with GCP is easy. You can create a free account and start using GCP services immediately.

To create a GCP account, visit the GCP website and click on the "Create an account" button. You will need to provide your name, email address, and a password.

Once you have created an account, you can start using GCP services. To learn more about GCP, you can visit the GCP documentation website.

Common use cases for GCP

GCP can be used for a wide variety of applications, including:

- Web and mobile applications: GCP can be used to host web and mobile applications.
- Data storage and analytics: GCP can be used to store and analyse data.
- Machine learning and artificial intelligence: GCP can be used to develop and deploy machine learning and artificial intelligence models.

3.STEPS:

REAL-TIME STOCK PRICE MONITORING

Build a real-time stock price monitoring system that alerts users when a stock price crosses a certain threshold. Use Cloud Pub/Sub to stream real-time stock price updates and Cloud Functions to analyse and send alerts.

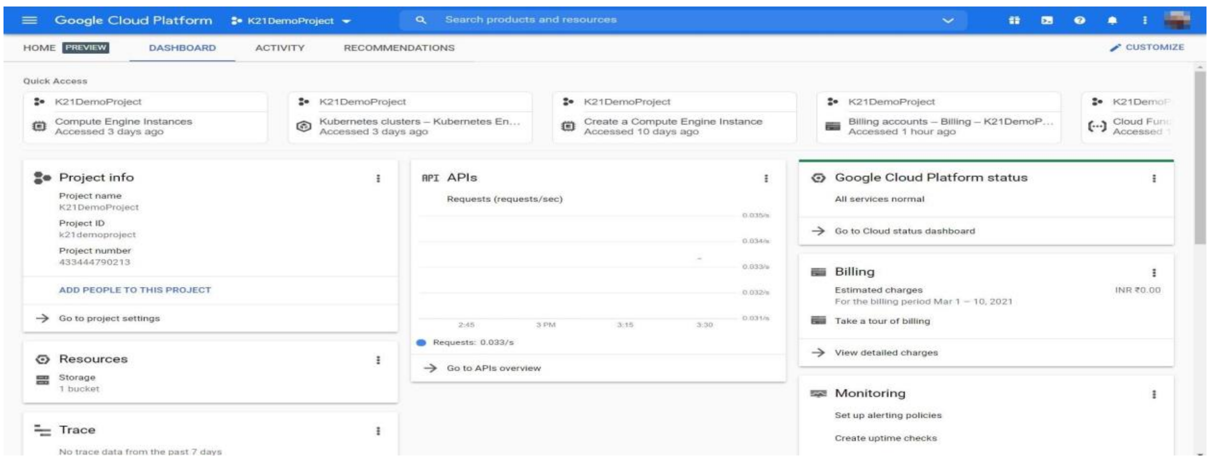
Steps:

- Fetching the stock price of Infosys by parsing the price html class and pushing it into up Cloud Pub/Sub.
- Used cloud function to analyse whether the stock price have raised above the threshold value.
- Send alerts via email by Cloud Function.

4. PLATFORM USED:

GOOGLE CLOUD PLATFORM INTERFACE:

Web Console:



Quick Access

K21DemoProject
Compute Engine Instances
Accessed 3 days ago

K21DemoProject
Kubernetes clusters – Kubernetes En...
Accessed 3 days ago

K21DemoProject
Create a Compute Engine Instance
Accessed 10 days ago

K21DemoProject
Billing accounts – Billing – K21DemoP...
Accessed 1 hour ago

K21DemoP
Cloud Fun
Accessed

Project info
Project name
K21DemoProject
Project ID
k21demoproject
Project number
433444790213
ADD PEOPLE TO THIS PROJECT
Go to project settings

Resources
Storage
1 bucket

Trace
No trace data from the past 7 days

APIs
Requests (requests/sec)
0.035/s
0.034/s
0.033/s
0.032/s
0.031/s
2:45 3 PM 3:15 3:30
Requests: 0.033/s
Go to APIs overview

Google Cloud Platform status
All services normal
Go to Cloud status dashboard

Billing
Estimated charges INR ₹0.00
For the billing period Mar 1 – 10, 2021
Take a tour of billing
View detailed charges

Monitoring
Set up alerting policies
Create uptime checks

Google Cloud Platform (GCP) Account: Sign up for a GCP account if you don't have one already. This provides access to GCP's storage services, including Cloud Storage for creating buckets and managing data.

Google Colab:

Google Colab (short for Collaboratory) is a cloud based Jupyter notebook environment that allows you to write and execute Python code. It’s particularly useful for data science, machine learning, and collaborative projects.

5.PROJECT REQUIREMENTS

Cloud Pub/Sub:

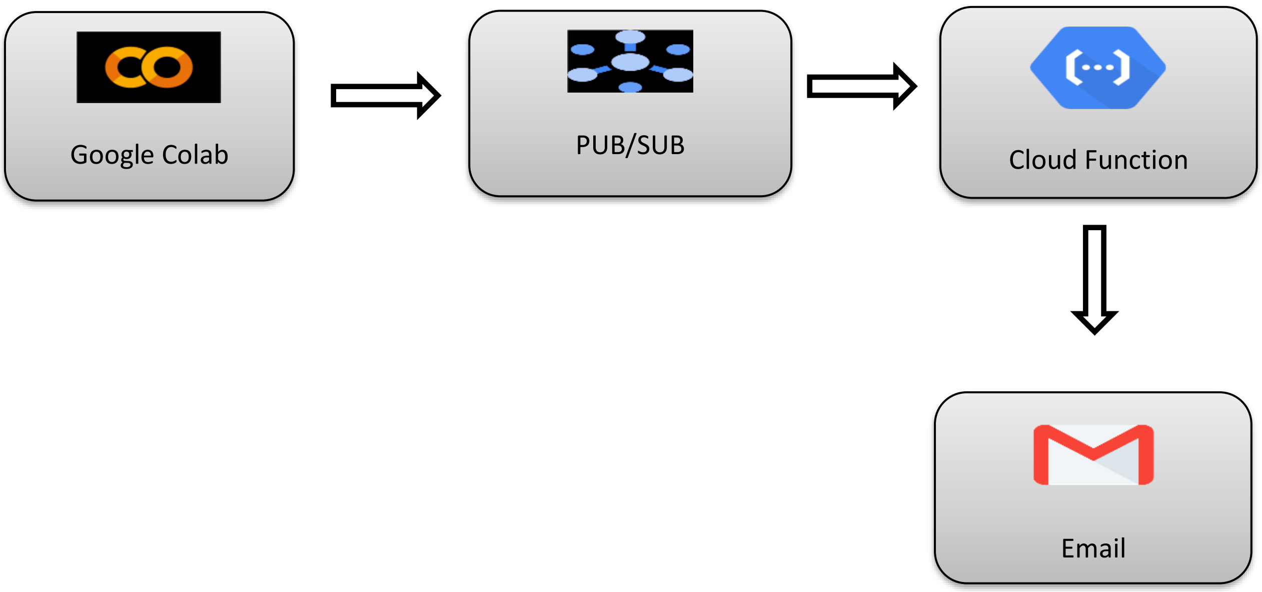
- Pub/Sub, short for Publisher/Subscriber is a messaging service provided by Google cloud platform (GCP) that enables asynchronous communication between independent application.
- Publishers send messages to a topic, and Subscribers receive those messages from the topic they are interested in.
- This decoupling allows for scalable and flexible communication between components of an application or between different applications altogether.

- Pub/Sub supports durable messaging, meaning messages are retained even if subscribers are temporarily offline, ensuring reliable delivery.
- Additionally, it integrates seamlessly with other GCP services, making it an essential tool for building distributed and event-driven architectures in the cloud.

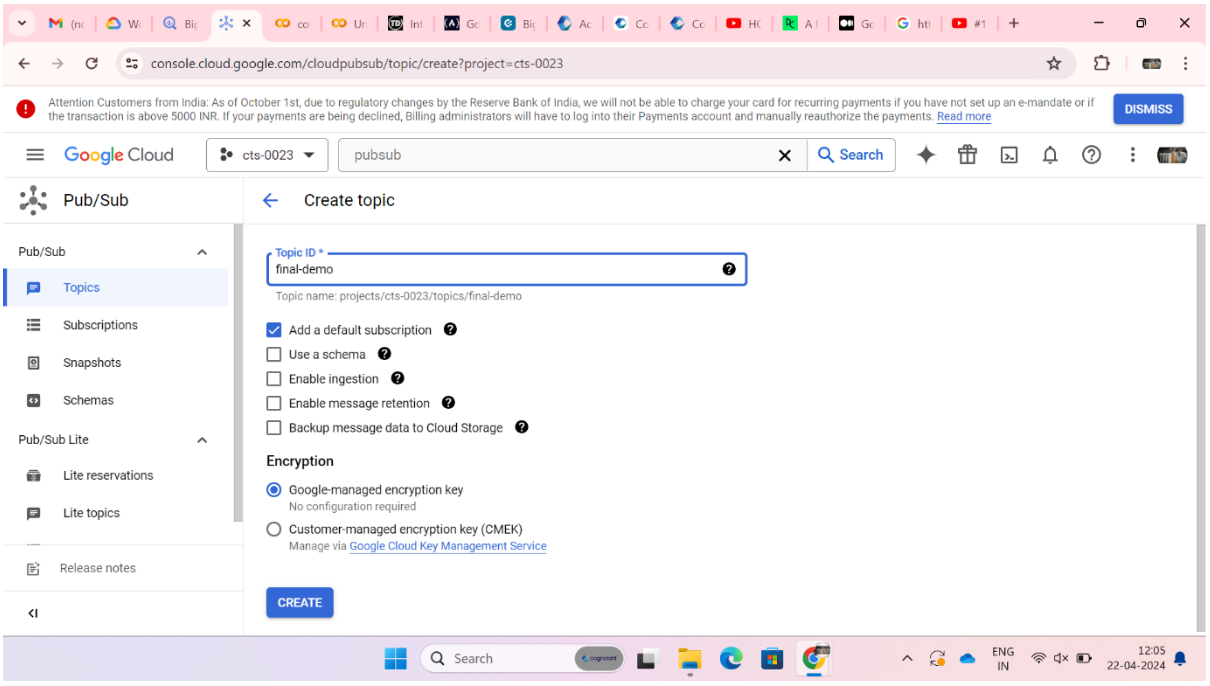
CLOUD FUNCTION:

- Cloud Functions is a serverless computing service provided by Google Cloud Platform (GCP) that allows developers to deploy and run event-driven functions in a fully managed environment.
- With Cloud Functions, developers can write lightweight code snippets in languages such as Node.js, Python, Go, and more, and deploy them without worrying about managing infrastructure.
- These functions are triggered by events from various GCP services, such as Cloud Storage, Cloud Pub/Sub, HTTP requests, and many others.
- Cloud Functions automatically scales up or down to match the incoming workload, ensuring optimal performance and cost-effectiveness.
- It is also called as a Function as a Service (FAAS).

6. IMPLEMENTATION



Firstly, we are creating a topic named as “final-demo” with default subscription.



console.cloud.google.com/cloudpubsub/topic/create?project=cts-0023

Attention Customers from India: As of October 1st, due to regulatory changes by the Reserve Bank of India, we will not be able to charge your card for recurring payments if you have not set up an e-mandate or if the transaction is above 5000 INR. If your payments are being declined, Billing administrators will have to log into their Payments account and manually reauthorize the payments. Read more

Create topic

Topic ID *: final-demo

Topic name: projects/cts-0023/topics/final-demo

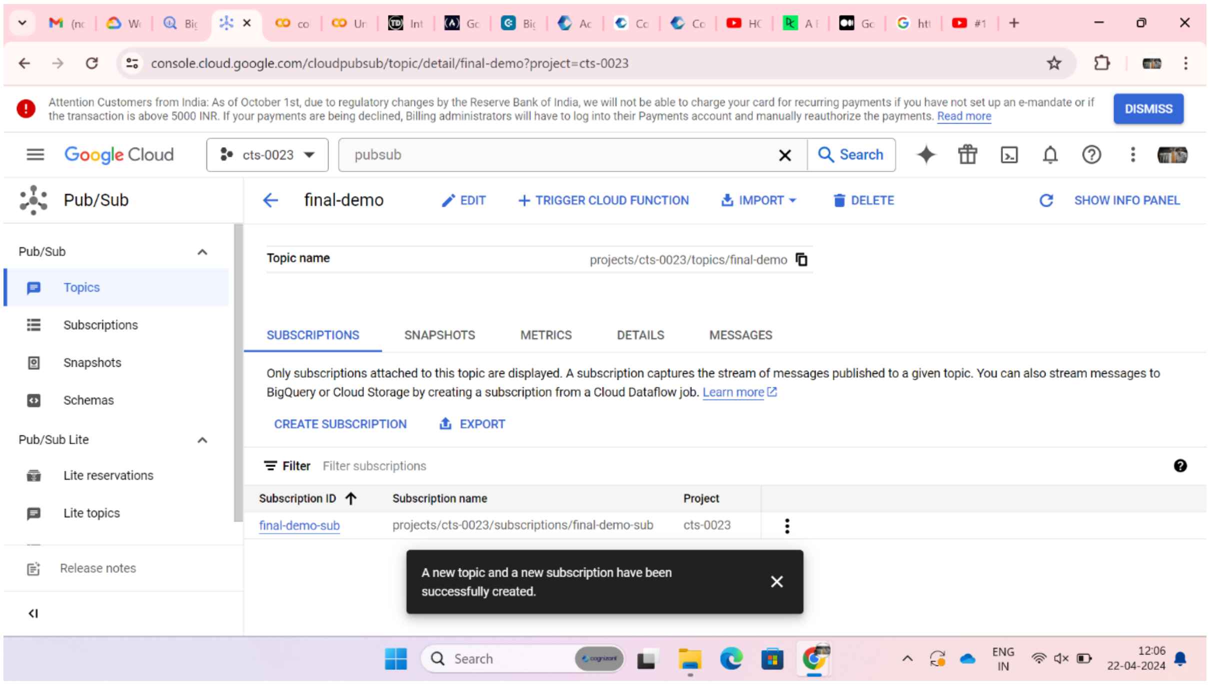
- [x] Add a default subscription
- [ ] Use a schema
- [ ] Enable ingestion
- [ ] Enable message retention
- [ ] Backup message data to Cloud Storage

Encryption

[✓ Google-managed encryption key] — No configuration required

Customer-managed encryption key (CMEK) — Manage via Google Cloud Key Management Service

Here, default subscription is named as “final-demo-sub”.



console.cloud.google.com/cloudpubsub/topic/detail/final-demo?project=cts-0023

Attention Customers from India: As of October 1st, due to regulatory changes by the Reserve Bank of India, we will not be able to charge your card for recurring payments if you have not set up an e-mandate or if the transaction is above 5000 INR. If your payments are being declined, Billing administrators will have to log into their Payments account and manually reauthorize the payments. Read more

final-demo

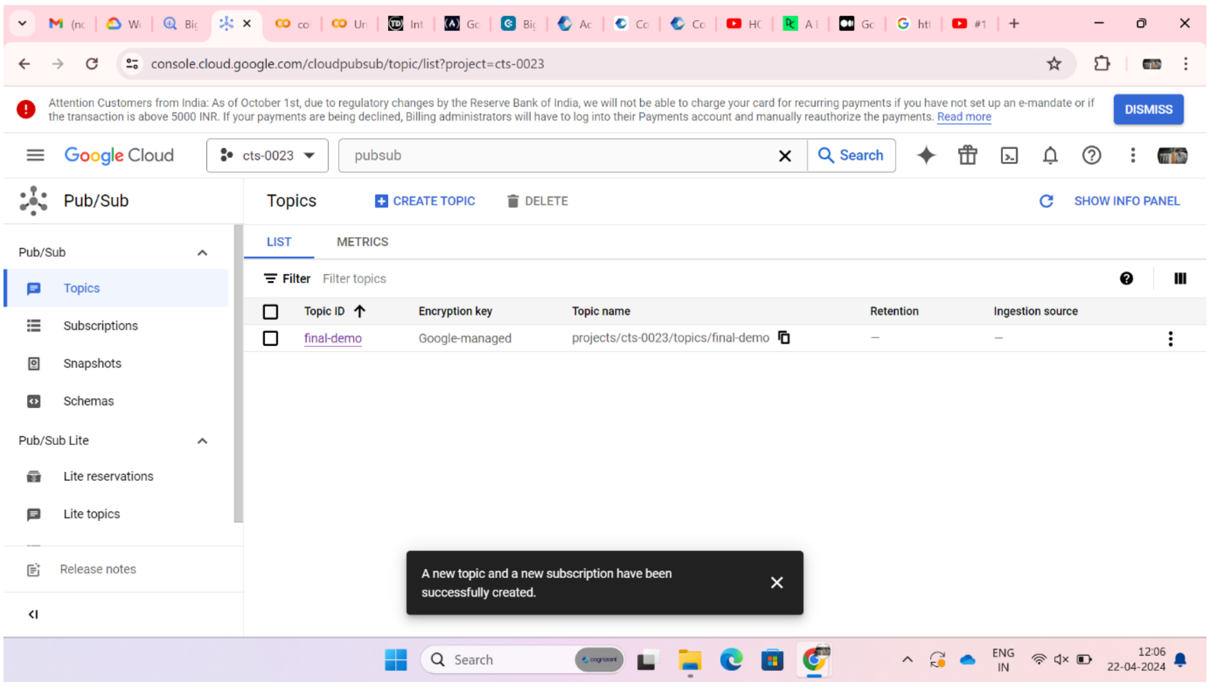
Topic name: projects/cts-0023/topics/final-demo

Only subscriptions attached to this topic are displayed. A subscription captures the stream of messages published to a given topic. You can also stream messages to BigQuery or Cloud Storage by creating a subscription from a Cloud Dataflow job. Learn more

Subscription ID	Subscription name	Project
final-demo-sub	projects/cts-0023/subscriptions/final-demo-sub	cts-0023

A new topic and a new subscription have been successfully created.

The topic named “final-demo” is created successfully.



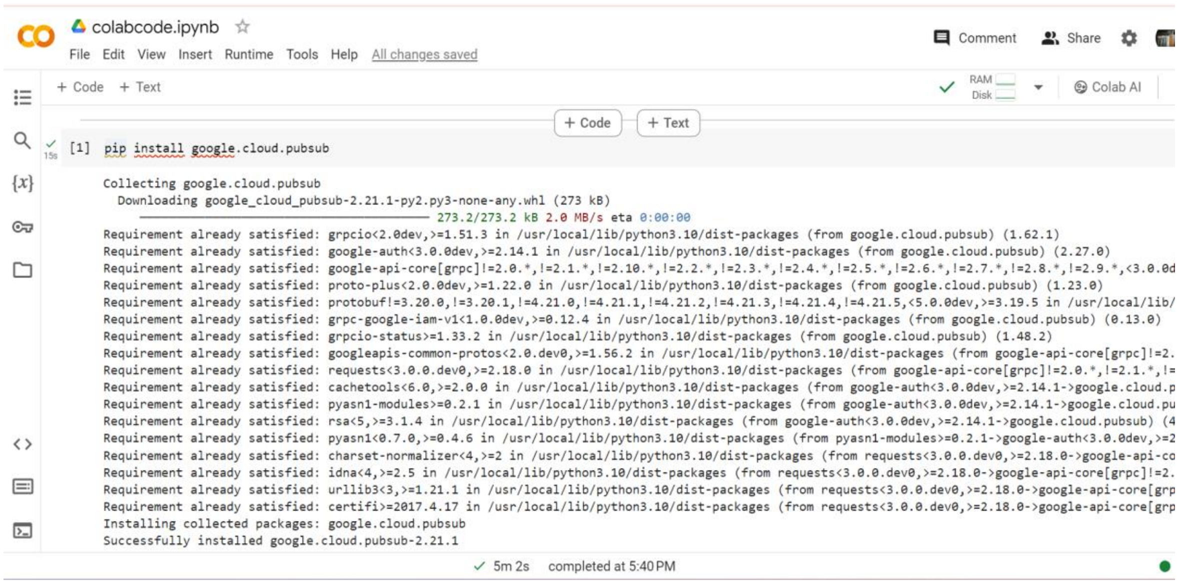
console.cloud.google.com/cloudpubsub/topic/list?project=cts-0023

Attention Customers from India: As of October 1st, due to regulatory changes by the Reserve Bank of India, we will not be able to charge your card for recurring payments if you have not set up an e-mandate or if the transaction is above 5000 INR. If your payments are being declined, Billing administrators will have to log into their Payments account and manually reauthorize the payments. Read more

Topic ID	Encryption key	Topic name	Retention	Ingestion source
final-demo	Google-managed	projects/cts-0023/topics/final-demo	–	–

A new topic and a new subscription have been successfully created.

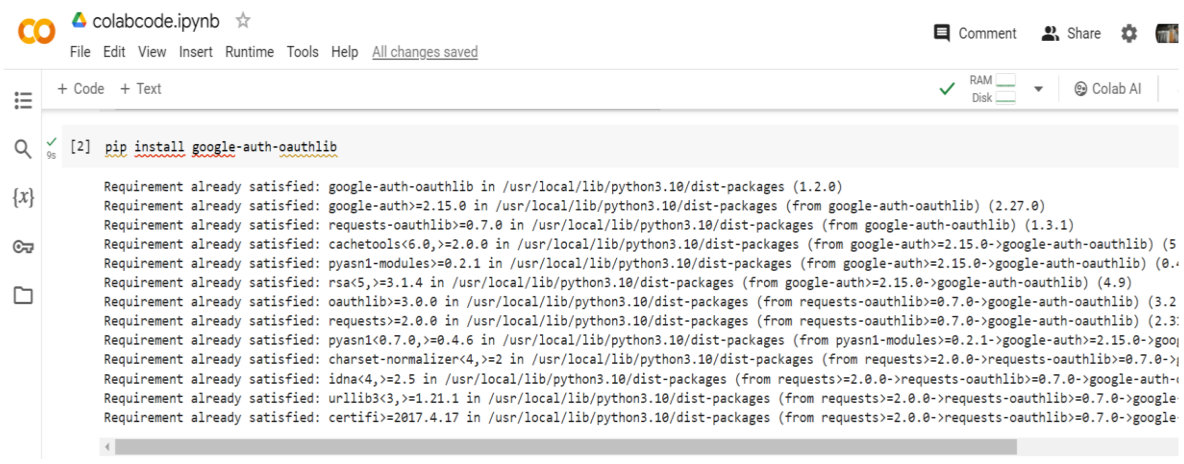
This line installs the `google.cloud.pubsub` package, which is required for interacting with Google Cloud Pub/Sub.



```
[1] pip install google.cloud.pubsub

    Collecting google.cloud.pubsub
      Downloading google_cloud_pubsub-2.21.1-py2.py3-none-any.whl (273 kB)
         ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 273.2/273.2 kB 2.0 MB/s eta 0:00:00
    Requirement already satisfied: grpcio<2.0dev,>=1.51.3 in /usr/local/lib/python3.10/dist-packages (from google.cloud.pubsub) (1.62.1)
    Requirement already satisfied: google-auth<3.0.0dev,>=2.14.1 in /usr/local/lib/python3.10/dist-packages (from google.cloud.pubsub) (2.27.0)
    Requirement already satisfied: google-api-core[grpc]!=2.0.*,!=2.1.*,!=2.10.*,!=2.2.*,!=2.3.*,!=2.4.*,!=2.5.*,!=2.6.*,!=2.7.*,!=2.8.*,!=2.9.*,<3.0.0d
    Requirement already satisfied: proto-plus<2.0.0dev,>=1.22.0 in /usr/local/lib/python3.10/dist-packages (from google.cloud.pubsub) (1.23.0)
    Requirement already satisfied: protobuf!=3.20.0,!=3.20.1,!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.19.5 in /usr/local/lib/
    Requirement already satisfied: grpc-google-iam-v1<1.0.0dev,>=0.12.4 in /usr/local/lib/python3.10/dist-packages (from google.cloud.pubsub) (0.13.0)
    Requirement already satisfied: grpcio-status>=1.33.2 in /usr/local/lib/python3.10/dist-packages (from google.cloud.pubsub) (1.48.2)
    Requirement already satisfied: googleapis-common-protos<2.0.dev0,>=1.56.2 in /usr/local/lib/python3.10/dist-packages (from google-api-core[grpc]!=2.
    Requirement already satisfied: requests<3.0.0.dev0,>=2.18.0 in /usr/local/lib/python3.10/dist-packages (from google-api-core[grpc]!=2.0.*,!=2.1.*,!=
    Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from google-auth<3.0.0dev,>=2.14.1->google.cloud.p
    Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from google-auth<3.0.0dev,>=2.14.1->google.cloud.pu
    Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.10/dist-packages (from google-auth<3.0.0dev,>=2.14.1->google.cloud.pubsub) (4
    Requirement already satisfied: pyasn1<0.7.0,>=0.4.6 in /usr/local/lib/python3.10/dist-packages (from pyasn1-modules>=0.2.1->google-auth<3.0.0dev,>=2
    Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0->google-api-co
    Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0->google-api-core[grpc]!=2.
    Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0->google-api-core[grp
    Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0->google-api-core[grp
    Installing collected packages: google.cloud.pubsub
    Successfully installed google.cloud.pubsub-2.21.1
```

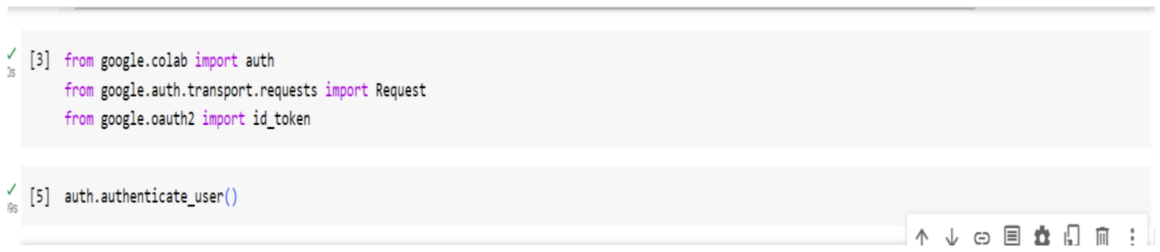
✓ 5m 2s completed at 5:40 PM

This line installs the `google-auth-oauthlib` package, which is required for handling authentication with Google services.



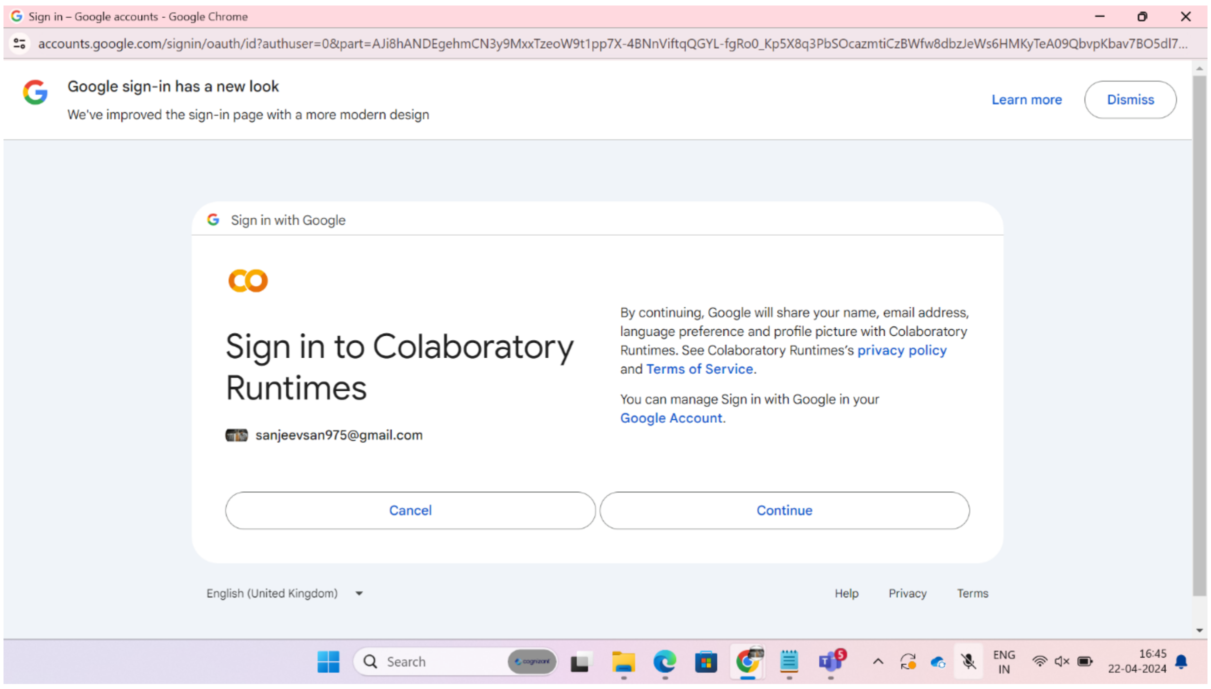
```
[2] pip install google-auth-oauthlib

Requirement already satisfied: google-auth-oauthlib in /usr/local/lib/python3.10/dist-packages (1.2.0)
Requirement already satisfied: google-auth>=2.15.0 in /usr/local/lib/python3.10/dist-packages (from google-auth-oauthlib) (2.27.0)
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from google-auth-oauthlib) (1.3.1)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from google-auth>=2.15.0->google-auth-oauthlib) (5
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from google-auth>=2.15.0->google-auth-oauthlib) (0.
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.10/dist-packages (from google-auth>=2.15.0->google-auth-oauthlib) (4.9)
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.10/dist-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib) (3.2
Requirement already satisfied: requests>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib) (2.3
Requirement already satisfied: pyasn1<0.7.0,>=0.4.6 in /usr/local/lib/python3.10/dist-packages (from pyasn1-modules>=0.2.1->google-auth>=2.15.0->goo
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests>=2.0.0->requests-oauthlib>=0.7.0->
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests>=2.0.0->requests-oauthlib>=0.7.0->google-auth-
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests>=2.0.0->requests-oauthlib>=0.7.0->google
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests>=2.0.0->requests-oauthlib>=0.7.0->google
```

- Here, it initiates the authentication process for the user, which is necessary for accessing Google Cloud services securely.



```
[3] from google.colab import auth
    from google.auth.transport.requests import Request
    from google.oauth2 import id_token

[5] auth.authenticate_user()
```

Here, Sign in page of colab will pop up as a part of Authentication from the OAuth package.


Sign in – Google accounts - Google Chrome

accounts.google.com/signin/oauth/id?authuser=0&part=AJi8hANDEgehmCN3y9MxxTzeoW9t1pp7X-4BNnViftqQGYL-fgRo0_Kp5X8q3PbSOcazmtiCzBWfw8dbzJeWs6HMKyTeA09QbvpKbav7BO5dl7...

Google sign-in has a new look

We've improved the sign-in page with a more modern design

Sign in with Google

Sign in to Colaboratory Runtimes

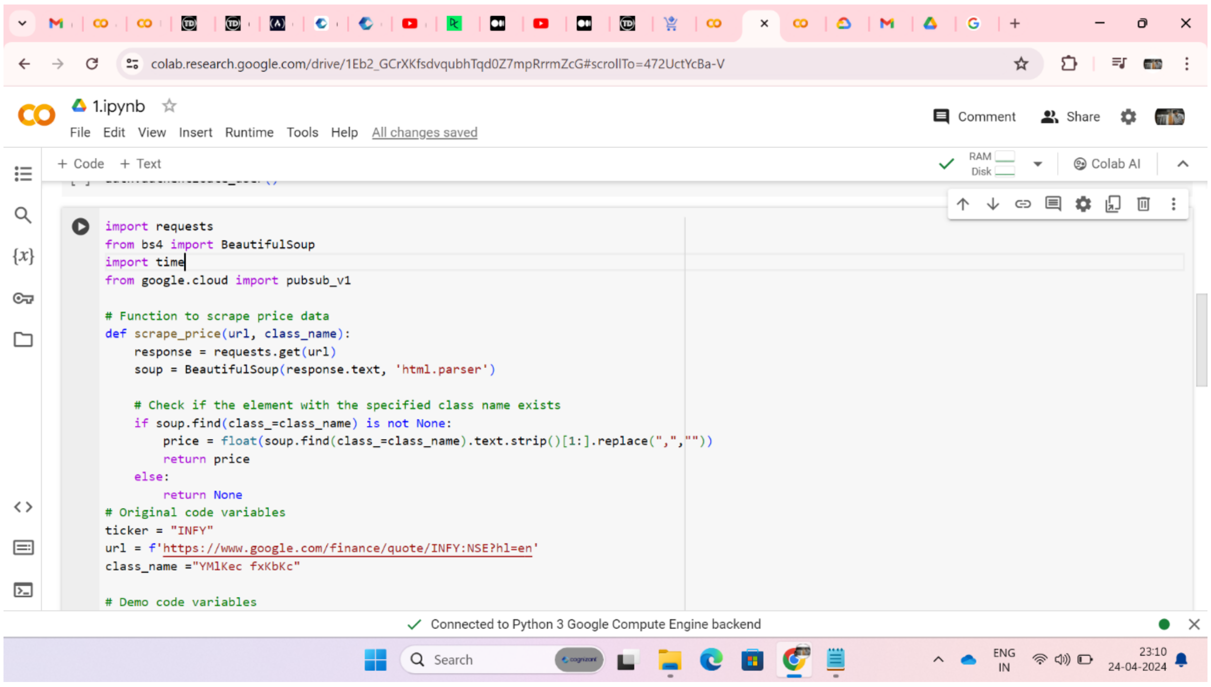
sanjeevsan975@gmail.com

By continuing, Google will share your name, email address, language preference and profile picture with Colaboratory Runtimes. See Colaboratory Runtimes's privacy policy and Terms of Service.

You can manage Sign in with Google in your Google Account.

- Next line imports the `requests` module, which is used for making HTTP requests to fetch web pages or interact with web APIs.
- Then next line imports the `BeautifulSoup` class from the `bs4` (Beautiful Soup 4) package, which is used for parsing HTML and XML documents.
- Next line imports the `time` module, which is used for adding time delays or getting current time information.
- Then next line imports the `pubsub_v1` module from the `google.cloud` package, which provides functionality for interacting with Google Cloud Pub/Sub.
- Then next line defines a function named `scrape_price` that takes two parameters: `url` and `class_name`. This function is used to scrape price data from a webpage.

- Next line extracts the price data from the HTML content, converts it to a floating-point number, and removes any extraneous characters or formatting.

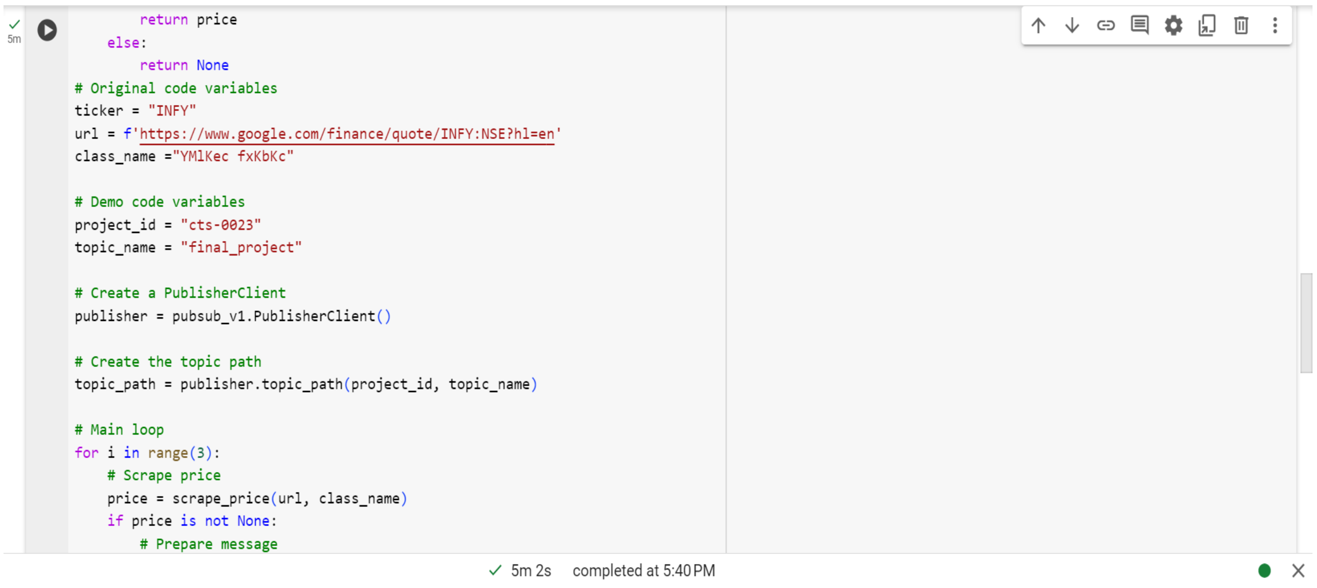


```python
import requests
from bs4 import BeautifulSoup
import time
from google.cloud import pubsub_v1

# Function to scrape price data
def scrape_price(url, class_name):
    response = requests.get(url)
    soup = BeautifulSoup(response.text, 'html.parser')

    # Check if the element with the specified class name exists
    if soup.find(class_=class_name) is not None:
        price = float(soup.find(class_=class_name).text.strip()[1:].replace(",",""))
        return price
    else:
        return None
# Original code variables
ticker = "INFY"
url = f'https://www.google.com/finance/quote/INFY:NSE?hl=en'
class_name ="YMlKec fxKbKc"

# Demo code variables
```

Connected to Python 3 Google Compute Engine backend

- Then it assigns the value `"INFY"` to the variable `ticker`, representing the stock ticker symbol.
- Then it constructs the URL for fetching the stock price data from Google Finance, using the ticker symbol `"INFY"`.
- Then it assigns the class name of the HTML element containing the stock price data to the variable `class_name`.
- Then next it creates a publisher client object for interacting with Cloud Pub/Sub.

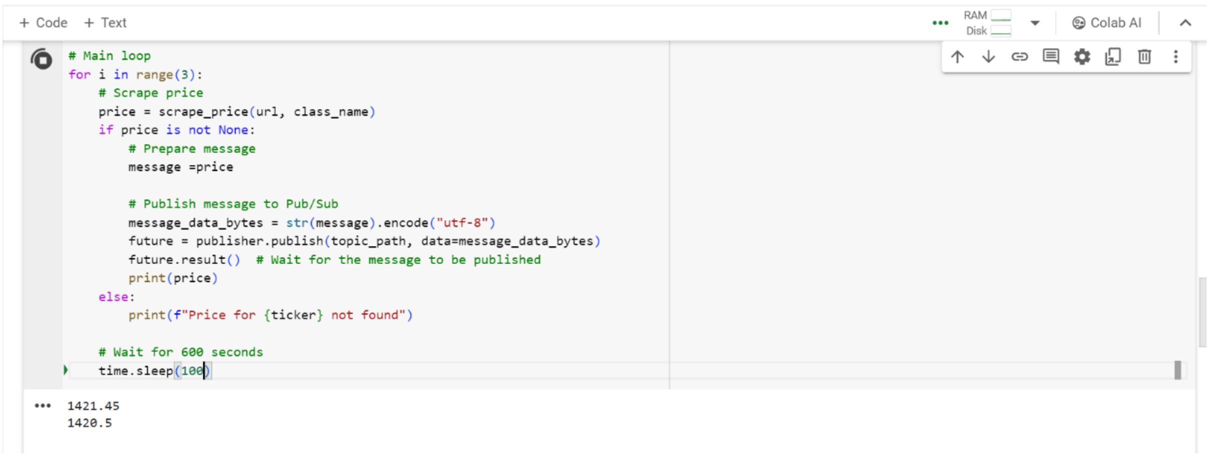
- Then it calls the `scrape_price` function to fetch the stock price data from the specified URL using the given class name.



```
        return price
    else:
        return None
# Original code variables
ticker = "INFY"
url = f'https://www.google.com/finance/quote/INFY:NSE?hl=en'
class_name ="YMlKec fxKbKc"

# Demo code variables
project_id = "cts-0023"
topic_name = "final_project"

# Create a PublisherClient
publisher = pubsub_v1.PublisherClient()

# Create the topic path
topic_path = publisher.topic_path(project_id, topic_name)

# Main loop
for i in range(3):
    # Scrape price
    price = scrape_price(url, class_name)
    if price is not None:
        # Prepare message
```

✓ 5m 2s completed at 5:40 PM

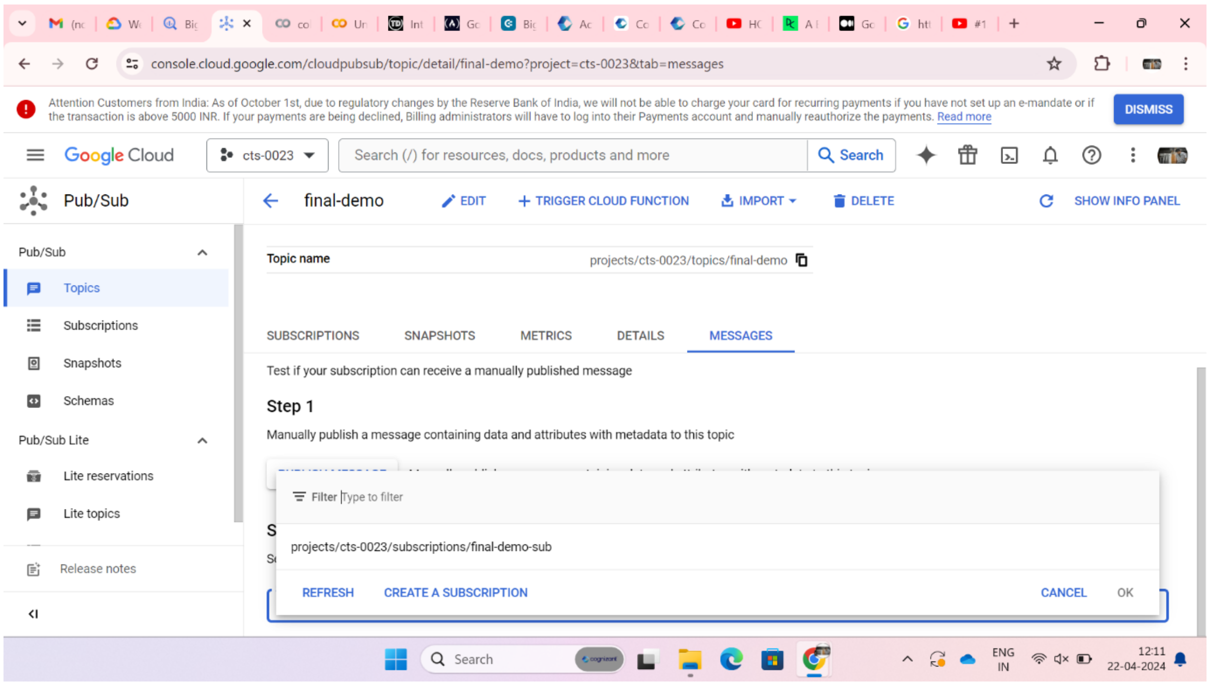
- The line encodes the message data as bytes before publishing it to Cloud Pub/Sub.
- Then next it publishes the message data to the Cloud Pub/Sub topic.
- Then next it waits for the message to be published before proceeding.
- Then it pauses the execution of the program for 100 seconds before the next iteration of the loop. This is used to limit the frequency of price updates to once every 100 seconds.



```
# Main loop
for i in range(3):
    # Scrape price
    price = scrape_price(url, class_name)
    if price is not None:
        # Prepare message
        message =price

        # Publish message to Pub/Sub
        message_data_bytes = str(message).encode("utf-8")
        future = publisher.publish(topic_path, data=message_data_bytes)
        future.result()  # Wait for the message to be published
        print(price)
    else:
        print(f"Price for {ticker} not found")

    # Wait for 600 seconds
    time.sleep(100)
```

```
1421.45
1420.5
```

After fetching the stock price, we are selecting the subscription ID to where the message must be sent.



console.cloud.google.com/cloudpubsub/topic/detail/final-demo?project=cts-0023&tab=messages

Attention Customers from India: As of October 1st, due to regulatory changes by the Reserve Bank of India, we will not be able to charge your card for recurring payments if you have not set up an e-mandate or if the transaction is above 5000 INR. If your payments are being declined, Billing administrators will have to log into their Payments account and manually reauthorize the payments. Read more

Topic name projects/cts-0023/topics/final-demo

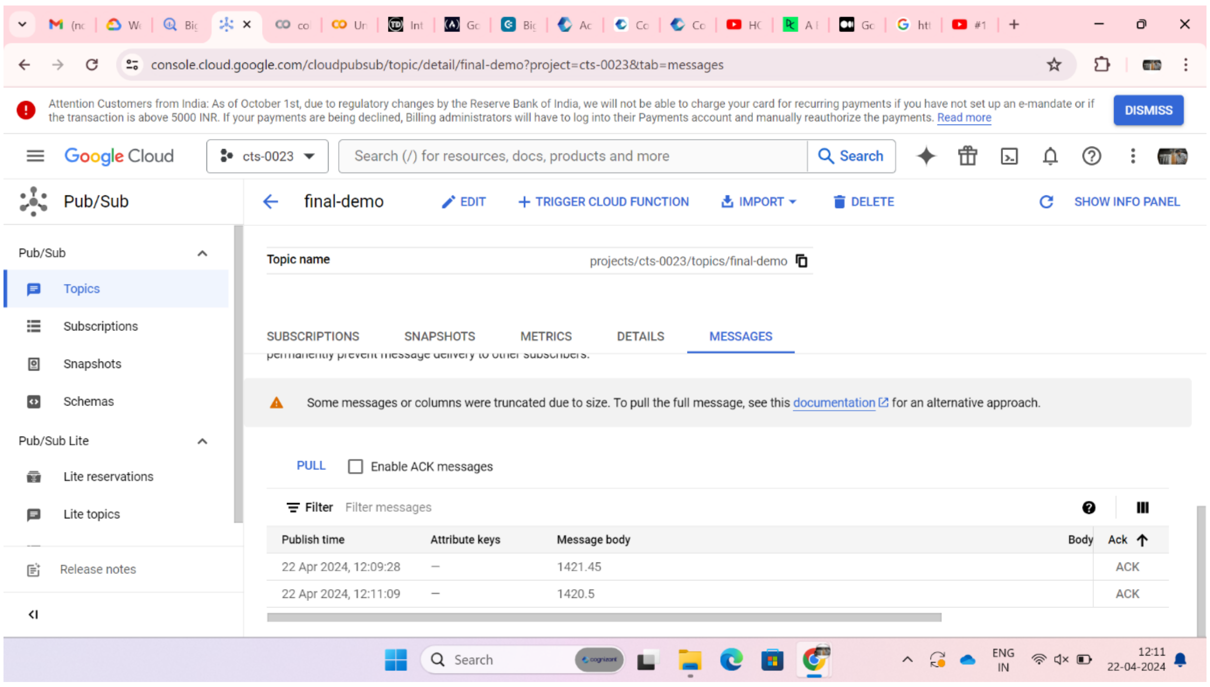
Test if your subscription can receive a manually published message

Step 1

Manually publish a message containing data and attributes with metadata to this topic

projects/cts-0023/subscriptions/final-demo-sub

The INFY stock price is published in topic messages.



console.cloud.google.com/cloudpubsub/topic/detail/final-demo?project=cts-0023&tab=messages

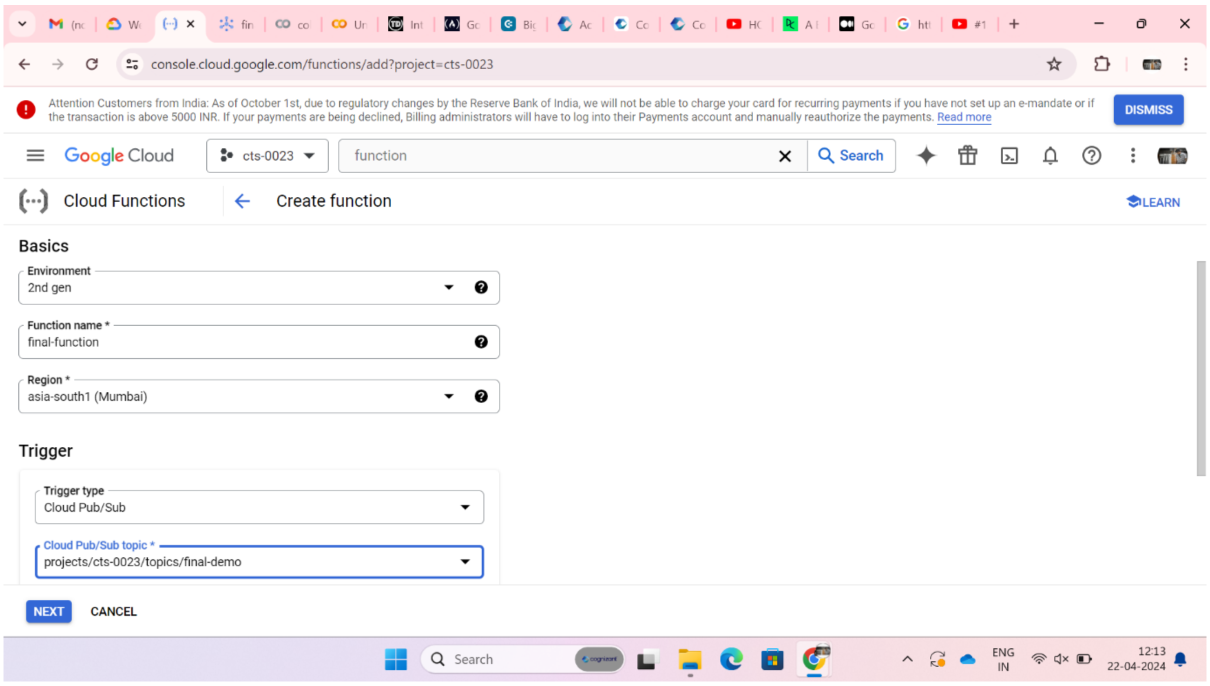
Attention Customers from India: As of October 1st, due to regulatory changes by the Reserve Bank of India, we will not be able to charge your card for recurring payments if you have not set up an e-mandate or if the transaction is above 5000 INR. If your payments are being declined, Billing administrators will have to log into their Payments account and manually reauthorize the payments. Read more

Topic name: projects/cts-0023/topics/final-demo

Some messages or columns were truncated due to size. To pull the full message, see this documentation for an alternative approach.

Publish time	Attribute keys	Message body	Ack
22 Apr 2024, 12:09:28	–	1421.45	ACK
22 Apr 2024, 12:11:09	–	1420.5	ACK

Then, we built a cloud function named “final-function” with trigger type “cloud pub/sub” with the respective topic.



console.cloud.google.com/functions/add?project=cts-0023

Attention Customers from India: As of October 1st, due to regulatory changes by the Reserve Bank of India, we will not be able to charge your card for recurring payments if you have not set up an e-mandate or if the transaction is above 5000 INR. If your payments are being declined, Billing administrators will have to log into their Payments account and manually reauthorize the payments. Read more

Basics

Environment: 2nd gen

Function name *: final-function

Region *: asia-south1 (Mumbai)

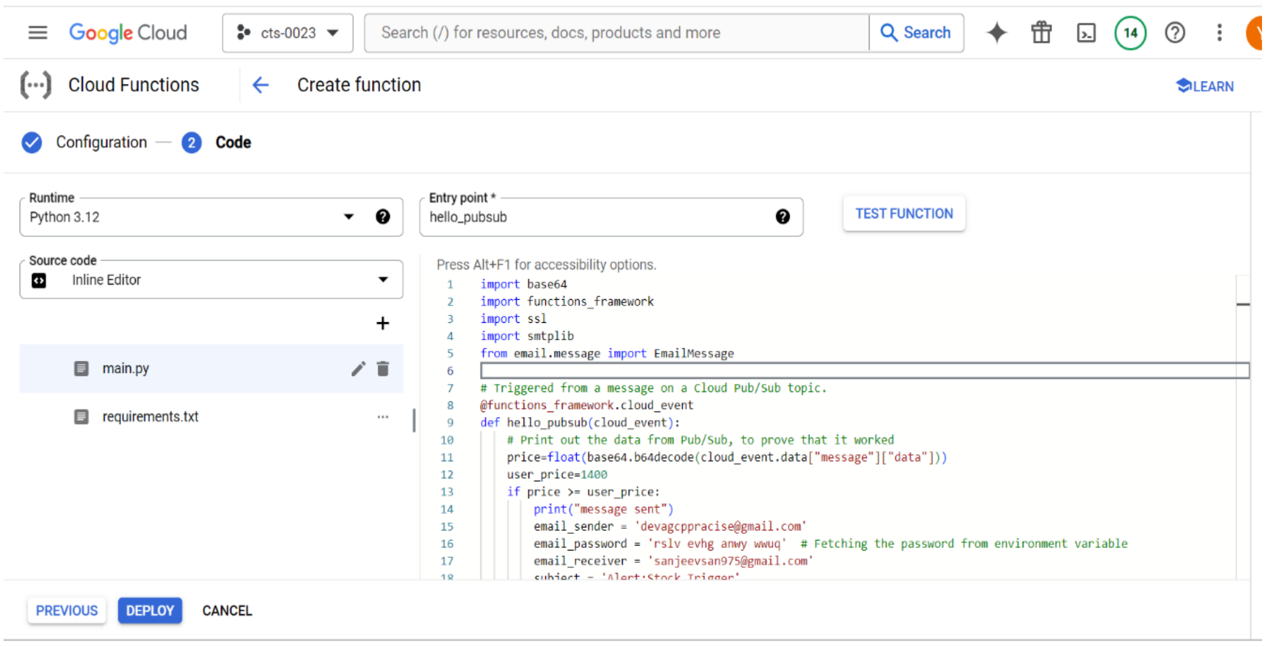
Trigger

Trigger type: Cloud Pub/Sub

Cloud Pub/Sub topic *: projects/cts-0023/topics/final-demo

- Here, we defined a cloud function .
- Initially, Base64 is a binary-to-text encoding scheme used to encode data.
- Functions_framework is a framework for writing lightweight functions as part of Google Cloud Functions

- Then we imported the EmailMessage class from the email.message module, which is used to represent an email message.
- Then it defines a function named `hello_pubsub` which takes a `cloud_event` as its parameter. This function will be triggered when a message is published to a Cloud Pub/Sub topic.



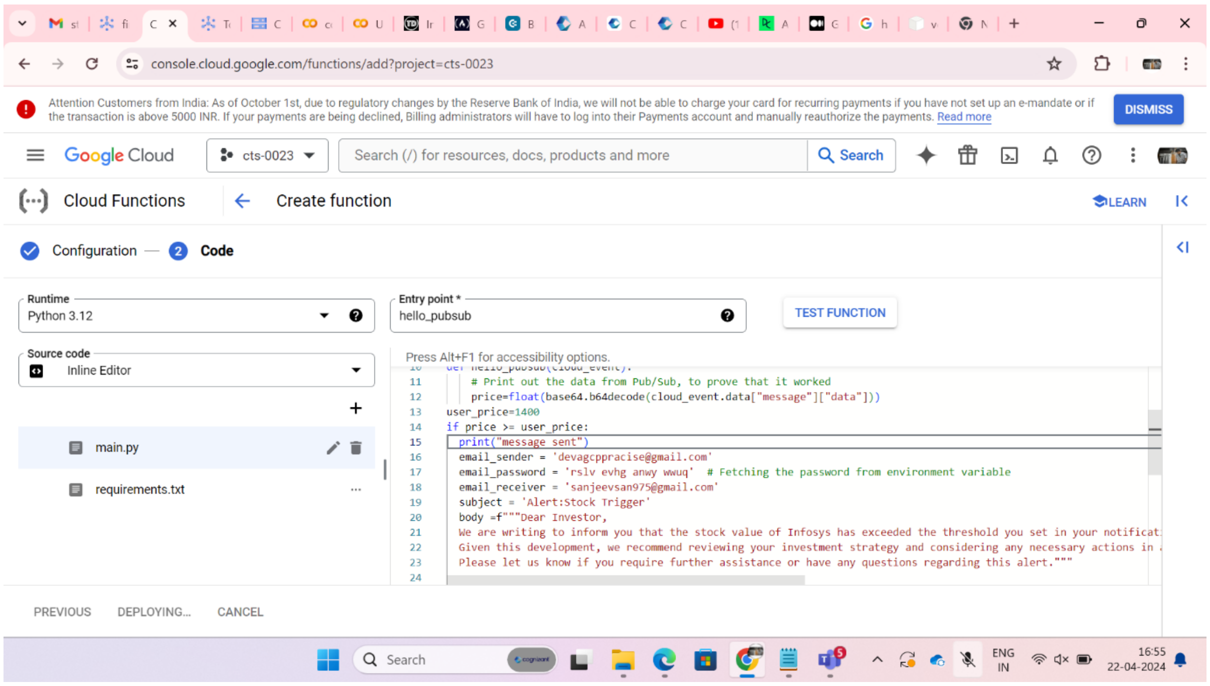
Runtime: Python 3.12
Entry point *: hello_pubsub
Source code: Inline Editor

main.py
requirements.txt

Press Alt+F1 for accessibility options.

```python
import base64
import functions_framework
import ssl
import smtplib
from email.message import EmailMessage

# Triggered from a message on a Cloud Pub/Sub topic.
@functions_framework.cloud_event
def hello_pubsub(cloud_event):
    # Print out the data from Pub/Sub, to prove that it worked
    price=float(base64.b64decode(cloud_event.data["message"]["data"]))
    user_price=1400
    if price >= user_price:
        print("message sent")
        email_sender = 'devagcppracise@gmail.com'
        email_password = 'rslv evhg anwy wwuq'  # Fetching the password from environment variable
        email_receiver = 'sanjeevsan975@gmail.com'
        subject = 'Alert:Stock Trigger'
```

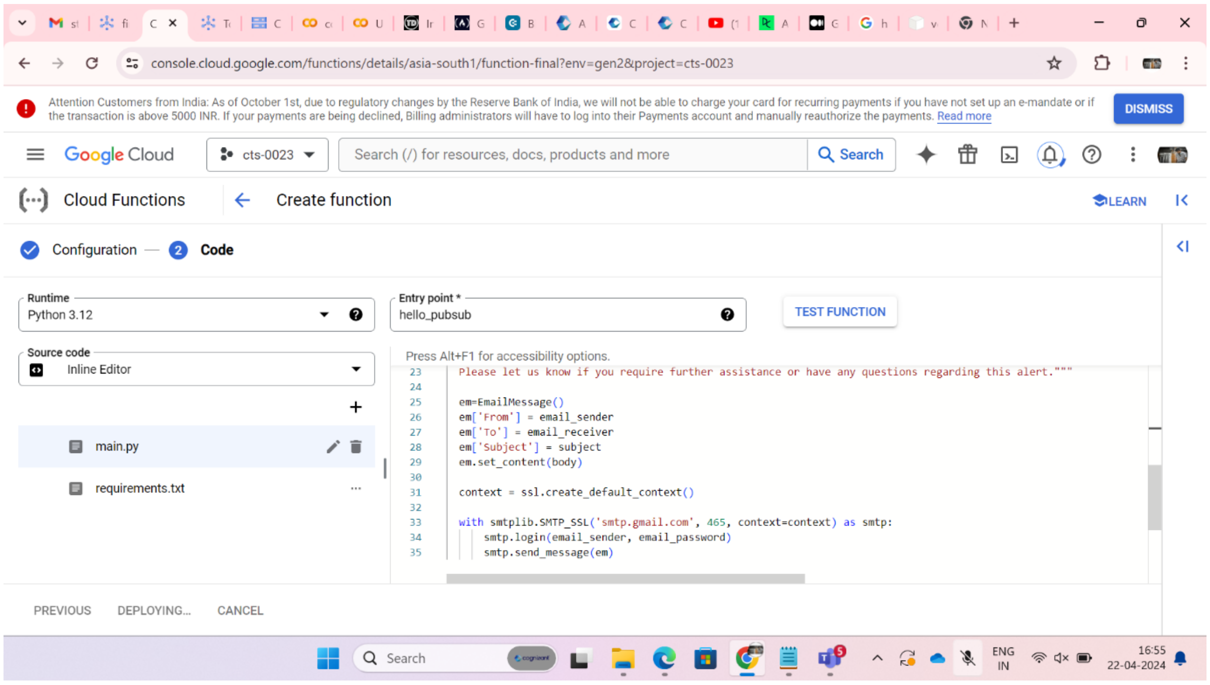
- This line decodes the data from the Cloud Pub/Sub message, which is encoded in base64 format. It then converts it to a float, assuming the data represents a numeric value like a stock price.
- Next it checks if the price extracted from the Pub/Sub message is greater than or equal to a threshold price (`user_price`). If the condition is true, it proceeds to send an alert.



console.cloud.google.com/functions/add?project=cts-0023
Attention Customers from India: As of October 1st, due to regulatory changes by the Reserve Bank of India, we will not be able to charge your card for recurring payments if you have not set up an e-mandate or if the transaction is above 5000 INR. If your payments are being declined, Billing administrators will have to log into their Payments account and manually reauthorize the payments. Read more
DISMISS
Google Cloud
cts-0023
Search (/) for resources, docs, products and more
Search
Cloud Functions
Create function
LEARN
Configuration
Code
PREVIOUS
DEPLOYING...
CANCEL

Runtime: Python 3.12
Entry point *: hello_pubsub
TEST FUNCTION
Source code: Inline Editor
main.py
requirements.txt
Press Alt+F1 for accessibility options.

```
def hello_pubsub(cloud_event):
    # Print out the data from Pub/Sub, to prove that it worked
    price=float(base64.b64decode(cloud_event.data["message"]["data"]))
user_price=1400
if price >= user_price:
 print("message sent")
 email_sender = 'devagcppracise@gmail.com'
 email_password = 'rslv evhg anwy wwuq'  # Fetching the password from environment variable
 email_receiver = 'sanjeevsan975@gmail.com'
 subject = 'Alert:Stock Trigger'
 body =f"""Dear Investor,
 We are writing to inform you that the stock value of Infosys has exceeded the threshold you set in your notificat
 Given this development, we recommend reviewing your investment strategy and considering any necessary actions in a
 Please let us know if you require further assistance or have any questions regarding this alert."""
```

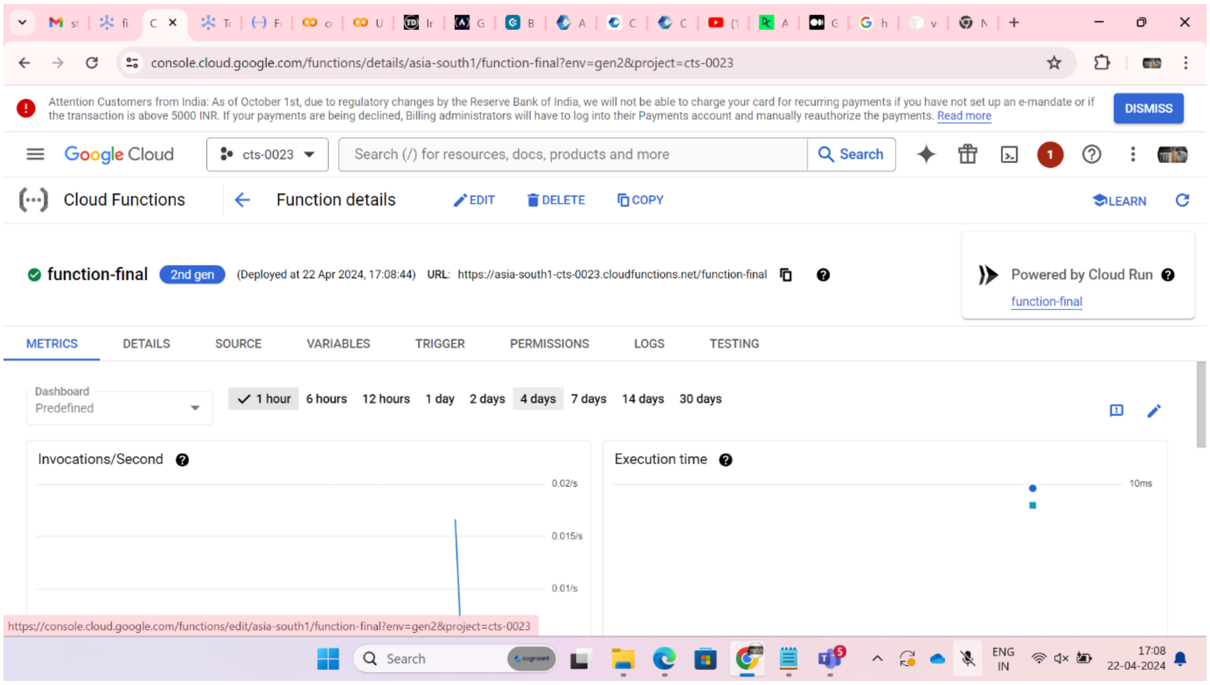


console.cloud.google.com/functions/details/asia-south1/function-final?env=gen2&project=cts-0023
Attention Customers from India: As of October 1st, due to regulatory changes by the Reserve Bank of India, we will not be able to charge your card for recurring payments if you have not set up an e-mandate or if the transaction is above 5000 INR. If your payments are being declined, Billing administrators will have to log into their Payments account and manually reauthorize the payments. Read more
DISMISS
Google Cloud
cts-0023
Search (/) for resources, docs, products and more
Search
Cloud Functions
Create function
LEARN
Configuration
Code
PREVIOUS
DEPLOYING...
CANCEL

Runtime: Python 3.12
Entry point *: hello_pubsub
TEST FUNCTION
Source code: Inline Editor
main.py
requirements.txt
Press Alt+F1 for accessibility options.

```
 Please let us know if you require further assistance or have any questions regarding this alert."""

 em=EmailMessage()
 em['From'] = email_sender
 em['To'] = email_receiver
 em['Subject'] = subject
 em.set_content(body)

 context = ssl.create_default_context()

 with smtplib.SMTP_SSL('smtp.gmail.com', 465, context=context) as smtp:
     smtp.login(email_sender, email_password)
     smtp.send_message(em)
```

The cloud function got deployed.



console.cloud.google.com/functions/details/asia-south1/function-final?env=gen2&project=cts-0023

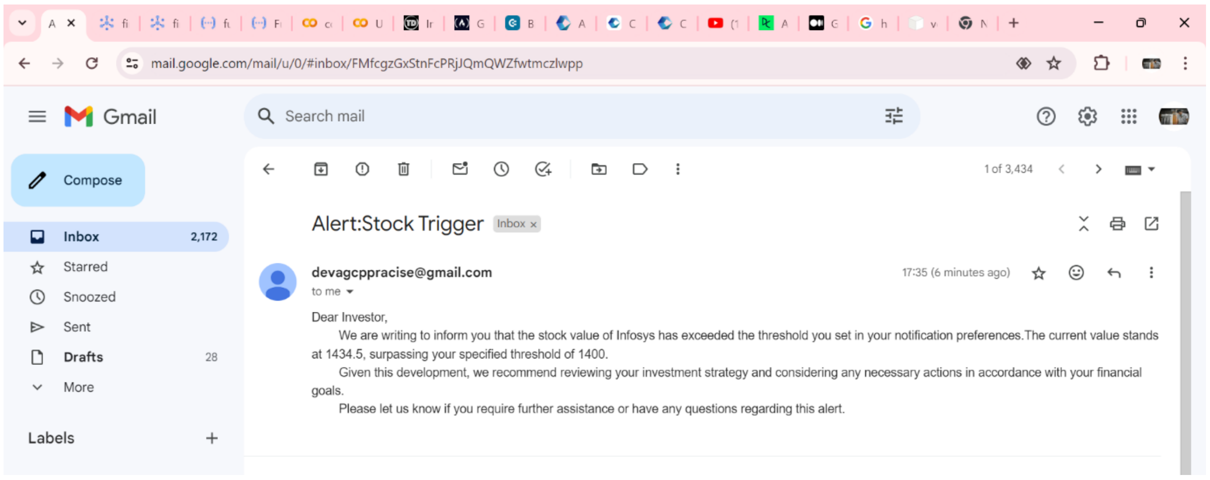
Attention Customers from India: As of October 1st, due to regulatory changes by the Reserve Bank of India, we will not be able to charge your card for recurring payments if you have not set up an e-mandate or if the transaction is above 5000 INR. If your payments are being declined, Billing administrators will have to log into their Payments account and manually reauthorize the payments. Read more

Cloud Functions — Function details

function-final 2nd gen (Deployed at 22 Apr 2024, 17:08:44) URL: https://asia-south1-cts-0023.cloudfunctions.net/function-final

Powered by Cloud Run function-final

When the stock price exceeded the threshold value then alert mail got triggered with a message.



mail.google.com/mail/u/0/#inbox/FMfcgzGxStnFcPRjJQmQWZfwtmczlwpp

Alert:Stock Trigger Inbox

devagcppracise@gmail.com
to me
17:35 (6 minutes ago)

Dear Investor,
We are writing to inform you that the stock value of Infosys has exceeded the threshold you set in your notification preferences.The current value stands at 1434.5, surpassing your specified threshold of 1400.
Given this development, we recommend reviewing your investment strategy and considering any necessary actions in accordance with your financial goals.
Please let us know if you require further assistance or have any questions regarding this alert.

7.CONCLUSION

- The real-time stock price monitoring system successfully scrapes stock prices, publishes them to Pub/Sub, and triggers alerts. With the suggested enhancements, you can make it even more powerful and user-friendly.
- We built a real-time stock price monitoring system that uses Cloud Pub/Sub to stream stock price updates and Cloud Functions to analyze and send alerts.
- The system scrapes stock price data from a website, publishes it to a Pub/Sub topic, and triggers a Cloud Function.
- Scraping component scrapes stock price data from a specified URL using BeautifulSoup. We extract the stock price from the HTML page.
- Cloud Function component subscribes to the Pub/Sub topic. It analyzes the stock price data and sends alerts (email and SMS) if the stock price crosses a certain threshold.