A

User Guide

ON

Graph E-Mail API

(Version-4.1.1)

Developed By

Ashad Khira

Ahmedabad – India

TABLE OF CONTENTS

1. Introduction

1.1 Project Profile

- In this section, we will discuss the profile of the project, The project is typically based on Globel Mail API that takes some user input & generates the Mail draft that displays graphical data analysis this is a fully automated mail system with chunks of inputs
- In this project we have an API that accepts Pull requests & returns the Mail in response
- In this API we are about to pass the Post request that takes a Payload as user input, in the Post request we are having 2 different params that work differently.
- Here are the four different endpoints with their specific URL
    - http://192.168.0.39:5020/composite_graph/
    - http://192.168.0.39:5020/comparison_graph/
    - http://192.168.0.39:5020/delta_table /
    - http://192.168.0.39:5020/exclusive_graph/
- The above URLs have below different params
    - Composite graph
        - The composite graph will take multiple files/tables from a user that works on multiple files & generate the graph as per inputs
    - Comparison graph
        - The comparison graph will take one file with a multiple header option that is dynamic, the user will have to enter the headers of a particular file, and if the user leaves the headers empty then the graph considers all available headers in the file/table & generates the mail accordingly
    - Delta table
        - This table has a limited central table format, it will take count date-wise from the fixed table format of SQL only
    - Exclusive Graph
        - Now onwards this endpoint has no limits of use, it just needs to provide the DataFrame, the graph, and the table dynamically set along with the DataFrame. Also, many more Dynamic parameters are available, Let's understand briefly below using payload.

1.2 Core Components

- Framework: Python Fast API
- Backend script: Python 3.8
- Database: MySQL/MongoDB
- OS: Linux Ubuntu
- Deployment: Gunicorn service
- Webserver: nginx

2. Development

2.1 Payload

Here is a payload that we are taking from the user & generating a graph accordingly

```
payload = {"project_name":'Project_Name',
        "type": 'SQL',
        "host":'192.168.1.42',
        "username":'root',
        "password":'examle',
        "database_name":'upscal_delivery_mail',
        "today_table":'upscal_count_master',
        "yesterday_table":'',
        "label_name":'[]',
        "days_to_compare":"5",
        "dynamic_table":'no',
        "table_prefix":'productdata',
        "table_date_format":"%Y_%m_%d",
        "header_flag": "no",
        "header_arr":"[]",
        "file_name":'File_Name',
        "Client_name":'Client_Name',
        "Alert_code":"123",
        "graph_type":'bar', #TODO - Types of graph :: bar, pie & line
        "graph_flag":'yes',
        "to_arr" : "['ashad.khira.examle@gmail.com']",
        "cc_arr":"['ashad.khira.examle@gmail.com']",
        "bcc_arr": "[]",
        "extra_params_arr":"['File Delivery Path: <Your File Path here>']",
        "file_base64":'',
        "client_io_flag":'yes',

        }
```

- **Payload for Exclusive Graph:**

```
payload = {
   "df_main": json.loads(df_main.to_json(orient='records')), #TODO - Required
   "df_name":'Date-wise Count', #TODO - Required
   "df_header": json.loads(df_main.to_json(orient='records')), #TODO - Optional
   "df_header_name":'My Header Table', #TODO - Optional
   "label_arr": df_today.columns.to_list(), #TODO - Required
   "project_name": "Test-Project", #TODO - Required
   "client_name": "Ashad Khira", #TODO - Required
   "graph_flag": "Yes", #TODO - Required
   "client_io_flag": "No", #TODO - Required
   "Alert_code": "6969", #TODO - Required
   "graph_type": "bar", #TODO - Types = Bar, Line, Radar, Pie, Doughnut, Polar, Scatter,
Bubble, Radial Gauge / Meter, Speedometer, Box Plots / violin
   "to_arr": ['ashad.khira.examle@gmail.com'], #TODO - Optional
   "cc_arr": ['krushil.gajjar.examle@gmail.com'], #TODO - Optional
   "bcc_arr": '', #TODO - Optional
   "header_box_label": ['Today Ratio', 'Yesterday Ratio'], #TODO - Optional
   "header_box_value": ['99%', '100%'] #TODO - Optional
}

payload_2 = {
   "df_main": [json.loads(df_main.to_json(orient='records')),
json.loads(df_main.to_json(orient='records'))], #TODO - For multiple table & graph
   # "df_main": json.loads(df_main.to_json(orient='records')), #TODO - For single table &
graph
   "df_name":['Monthly Data Report', 'Monthly Data Report 2'], #TODO - For Multiple
table & graph
   # "df_name":'Monthly Data Report', #TODO - For single table & graph
   "label_arr": [df_today.columns.to_list(), df_today.columns.to_list()], #TODO - For
Multiple table & graph
   # "label_arr": df_today.columns.to_list(),#TODO - For Single table & graph
   "project_name": "Test-Project", #TODO - Required
   "client_name": "Ashad Khira", #TODO - Required
   "graph_flag": "Yes", #TODO - Required
   "client_io_flag": "No", #TODO - Required
   "Alert_code": "6969", #TODO - Required
   "graph_type": "bar", #TODO - Types = Bar, Line, Radar, Pie, Doughnut, Polar, Scatter,
Bubble, Radial Gauge / Meter, Speedometer, Box Plots / violin
   "to_arr": ['ashad.khira.examle@gmail.com'], #TODO - Optional
   "cc_arr": '', #TODO - Optional
   "bcc_arr": '', #TODO - Optional
   "header_box_label": ['Today Ratio', 'Yesterday Ratio'], #TODO - Optional
   "header_box_value": ['99%', '100%'], #TODO - Optional
   "extra_params_arr": "['Status - File Uploaded']",
}
```

df_main :- This param is specified for the main data frame that you are going to pass, also there is another option for multiple graphs & multiple tables, to achieve that just give multiple DataFrames in a list

df_name :- The name will be added as the title for your given DataFrame, as same as multiple DataFrame, the data frame should have multiple names in a list if more than one DataFrame is passed.

df_header:- The header table is now also dynamically set along with DataFrame

df_header_name:- The title should provide for the same

label_arr:- Now you have to provide the additional label for the graph purpose along with your DataFrame. As same as multiple DataFrame, the data frame should have multiple label_arr in a list if more than one data frame is passed.

project_name:- Same as per ordinary payload

client_name:- Same as per ordinary payload

graph_flag :- Same as per ordinary payload

client_io_flag :- Same as per ordinary payload

Alert_code :- Same as per ordinary payload

graph_type :- Same as per ordinary payload

to_arr, cc_arr, bcc_arr :- Same as per ordinary payload (Now CC & BCC will optional)

header_box_label, header_box_value:- This parameter specified for the box that is displayed on the head of mail, it should contain a label & value for the same

extra_params_arr:- The other message you can pass using the ‘extra_params_arr’ variable

2.2 Endpoints

In our central mail API, we have a Post method request that takes payload from the user,
In this URL we are passing the four endpoints for the different inputs & output.

I. http://192.168.0.39:5020/composite_graph/
II. http://192.168.0.39:5020/comparison_graph/
III. http://192.168.0.39:5020/delta_table/
IV. http://192.168.0.39:5020/ exclusive_graph/

As per above, we have the same URL with four different endpoints.

I. composite_ graph
II. comparison_graph
III. delta_table
IV. exclusive_table

2.3 Working Model

In Python, we simply request to the API & get the response as same as we send the post request in Python using the request module with the particular payload syntax
As the payload is given, we describe each payload's arguments below:

"project_name": - This parameter specifies the Project Name

"type": - This parameter specifies the Database which can be Mongo or SQL

"host": - This parameter specifies the host of the database

"username": - This parameter specifies the username of the database

"password": - This parameter specifies the username of the database

“database_name”: - This parameter specifies the Database Name

“today_table”: - This parameter specifies the today’s table that you want to graph.
“yesterday_table”: - This parameter specifies yesterday’s table which you want to need the graph.

“label_arr”: - This parameter specifies the label of files that require only Composite graph

“dynamic_table”: - This parameter specifies the flag for the table is known or not known that should be yes or no.

“table_prefix”: - This parameter specifies the table prefix if the dynamic table is “Yes”

“table_date_format”: - This parameter specifies the table date format if the dynamic table is “Yes”

“**header_flag**”: - This parameter specifies that if you want header-wise comparison even if in composites graph

“header_arr”: - This parameter specifies the headers array in only the comparison parameter of the API

“file_name”: - This parameter specifies the file name which keeps the subject in mail format

“Client_name”: - This parameter specifies the file Client name which keeps the Client Name in mail format

“Alert_code”: - This parameter specifies the Project/Alert code for a specific project

“graph_type”: - This parameter specifies the type of the graph which can be “bar”, “line”, “pie”

“graph_flag”: - This parameter specifies the flag that the graph is to be shown or not, it should be “yes” or “no”

“to_arr”: - This parameter specifies which person we should keep on the “TO” mail.

“cc_arr”: - This parameter specifies which person we should keep on the “CC” mail.

“bcc_arr”: - This parameter specifies which person we should keep on the “BCC” mail

“extra_params_arr”: - This parameter specifies the extra params or arguments that user wants to keep in the mail

“**file_base64**”: - From now onwards you can attach the file in the mail also using this param, this params accepts the file content the code for it is given below

“**client_io_flag**”: - Now giving Yes & no you can change ‘from’ Mail id (receiver’s mail ID)

(Note**: if you are giving data in this param you have to specify the file name in **file_name** params**)

Code for file_base64**:**

```
with open(<Your File Path>, 'rb') as file:
    # exs = file_path.split()
    file_data = file.read()
    file_base64 = base64.b64encode(file_data).decode('utf-8')
```

2.4 Addon

When sending client emails (with the flag client_io_flag: "Yes"), the email will be sent from the address alerts@examle.io to the designated recipients in the "To" and "CC" fields. However, the "BCC" field will exclude alerts@examle.io, and a copy of the email will instead be sent from a *@gmail.com address.

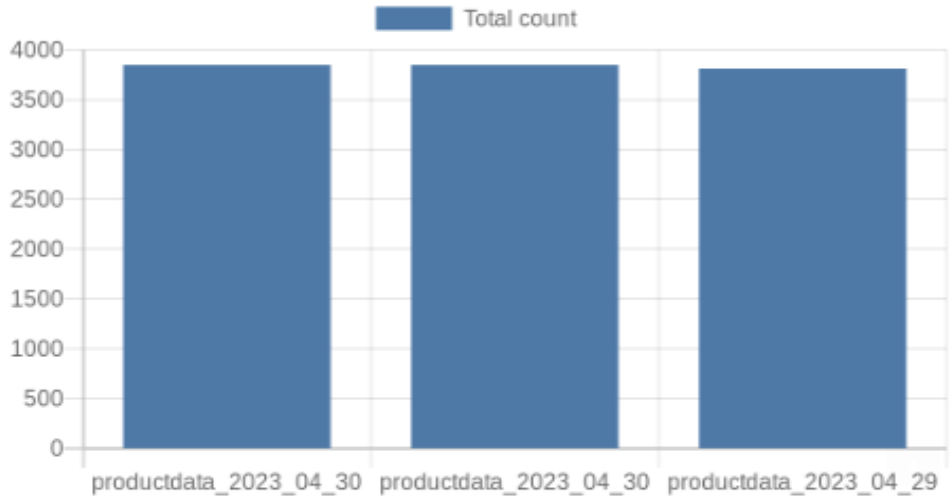
2.5 Working project screenshot

Composite Graph

nliq

nliq
24 May 2023

nliq



No	File Name	Total count
1	productdata_2023_04_30	3848
2	productdata_2023_04_30	3848
3	productdata_2023_04_29	3811

Remarks:

- Delivery_date : 22-05-2023
- AWS Path : Test/abc.zip

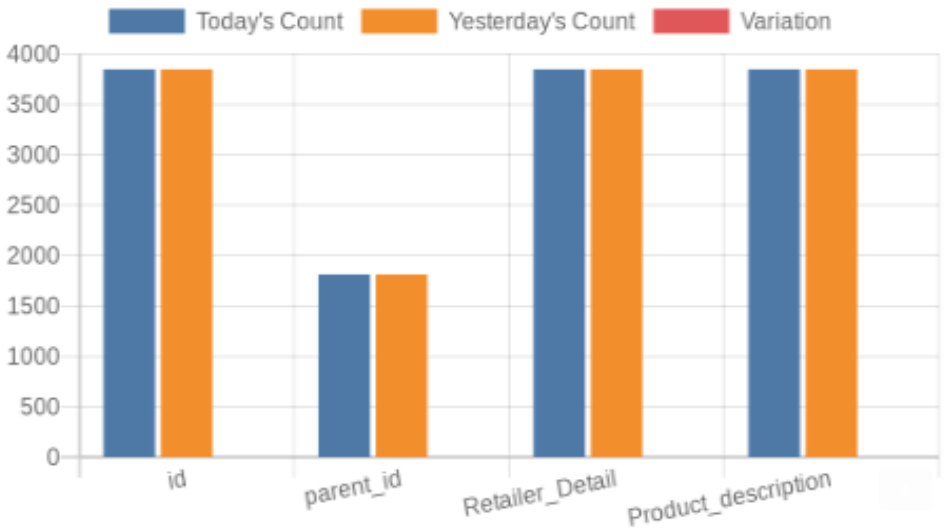
This is system generated mail - Do Not Reply

Comparison Graph

Project_Name

Client_Name
24 May 2023

File_Name



No	Header Name	Today's Count	Yesterday's Count	Variation
1	id	3848	3848	0
2	parent_id	1813	1813	0
3	Retailer_Detail	3848	3848	0
4	Product_description	3848	3848	0

Remarks:

- File Delivery Path: path/to/upload

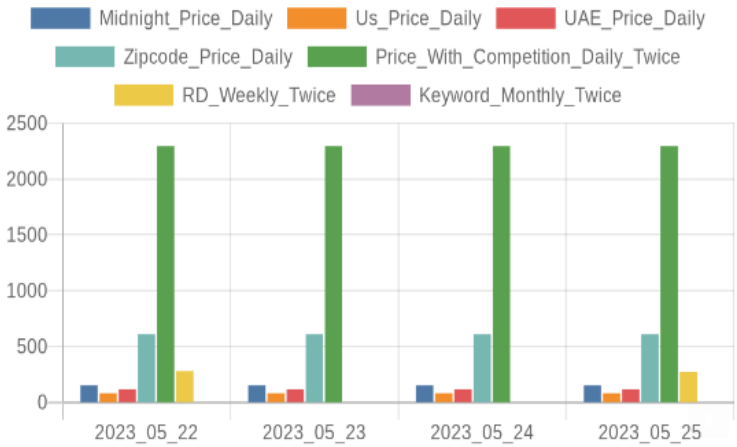
This is system generated mail - Do Not Reply

Delta Table Graph

Project_Name

Client_Name
26 May 2023

Project_Name



No	Date	Midnight_Price_Daily	Us_Price_Daily	UAE_Price_Daily	Zipcode_Price_Daily	Price_With_Competition_Daily_Twice	RD_Weekly_Twice	Keyword_Monthly_Twice
1	2023_05_22	152	80	116	608	2293	279	
2	2023_05_23	152	80	116	608	2293		
3	2023_05_24	152	80	116	608	2293		
4	2023_05_25	152	80	116	608	2293	272	

3. Conclusion

- In this project, we successfully developed the Central Mail API using FastAPI in Python, aiming to streamline and automate mail-related operations within our organization. The project objectives were to create a robust and efficient API that would handle various mail-related tasks.

4. Reference

- Herewith attached is a Dropbox folder link that contains two ".py" files that refer to the Post request to the API using the given Payload

    - **Comparison Graph Py file: -**
      https://www.dropbox.com/scl/fi/lm2mykp2yfjawd0hu15mj/api_comparison_graph_request.py?rlkey=yl8nzperaxpc007qike4ns2z6&dl=0

    - **Composite Graph Py file: -**
      https://www.dropbox.com/scl/fi/j14un9jqhwr06uuqo1wfi/api_composite_graph_request.py?rlkey=227ateidutiftpdo6y8qvdjyr&dl=0

    - **Delta Table Py file: -**
      https://www.dropbox.com/scl/fi/hdur4fso5e5do3vgdwa76/api_delta_table_request.py?rlkey=i4r15f1amjine10zjxumfaroa&dl=0

    - **Exclusive Table Py file: -**
      https://www.dropbox.com/scl/fi/rbw6wbld60mmwbzre6iop/api_exclusive_graph_request.py?rlkey=uiyugat6o5kb6slc1xen05wrk&dl=0