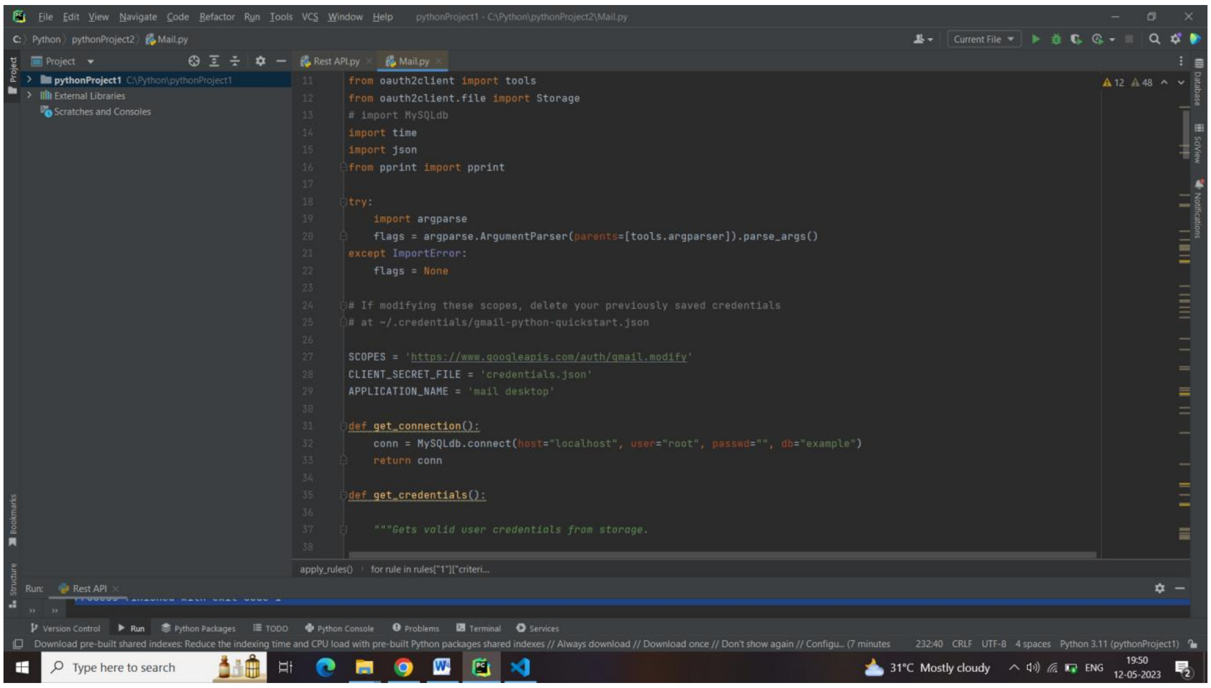
Write a standalone Python script that integrates with GMail API and performs some rule based operations on emails.

Summary of the project:

STEP 1. AUTHENTICATING GOOGLE’S GMAIL API

To authenticate my Gmail account with the Gmail API, follow the steps provided by Google to get an OAuth token. Use the official Gmail Python client library to interact with the Gmail API.

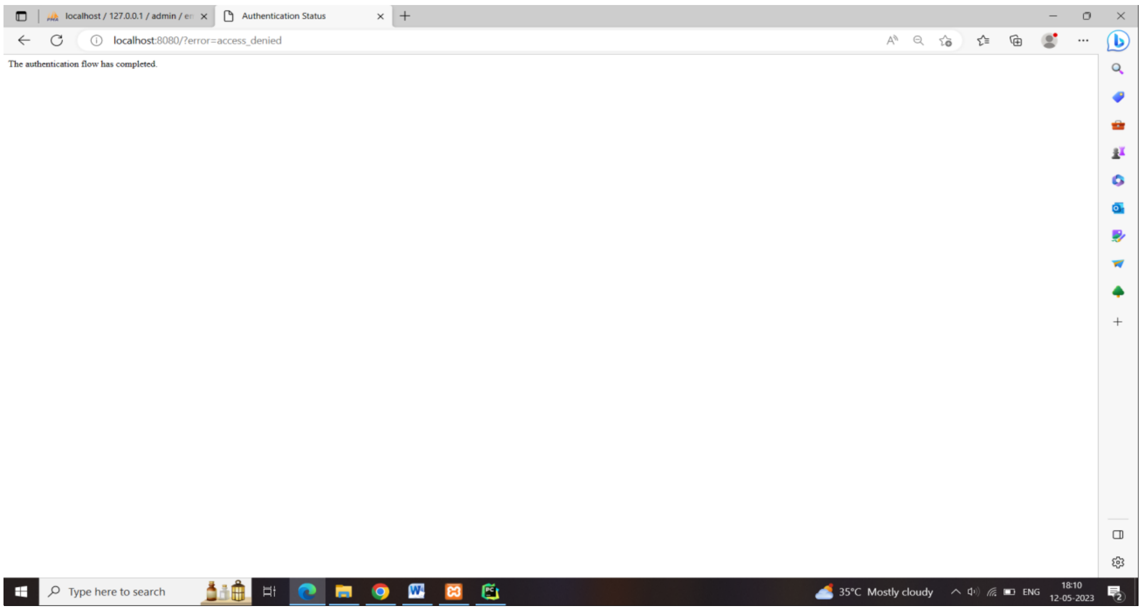
I Create a new folder to give the name Mail.py. This folder contains a code for authenticating to Google's Gmail API. In 10 th line adding scopes to complete the authentication process.



```python
from oauth2client import tools
from oauth2client.file import Storage
# import MySQLdb
import time
import json
from pprint import pprint

try:
    import argparse
    flags = argparse.ArgumentParser(parents=[tools.argparser]).parse_args()
except ImportError:
    flags = None

# If modifying these scopes, delete your previously saved credentials
# at ~/.credentials/gmail-python-quickstart.json

SCOPES = 'https://www.googleapis.com/auth/gmail.modify'
CLIENT_SECRET_FILE = 'credentials.json'
APPLICATION_NAME = 'mail desktop'

def get_connection():
    conn = MySQLdb.connect(host="localhost", user="root", passwd="", db="example")
    return conn

def get_credentials():

    """Gets valid user credentials from storage.

```

pythonProject1 - C:\Python\pythonProject2\Mail.py
Rest API.py
Mail.py
pythonProject1 C:\Python\pythonProject1
External Libraries
Scratches and Consoles
apply_rules() › for rule in rules["1"]["criteri...
Run: Rest API

STEP 2:OUTPUT OF AUTHENTICATION PROCESS

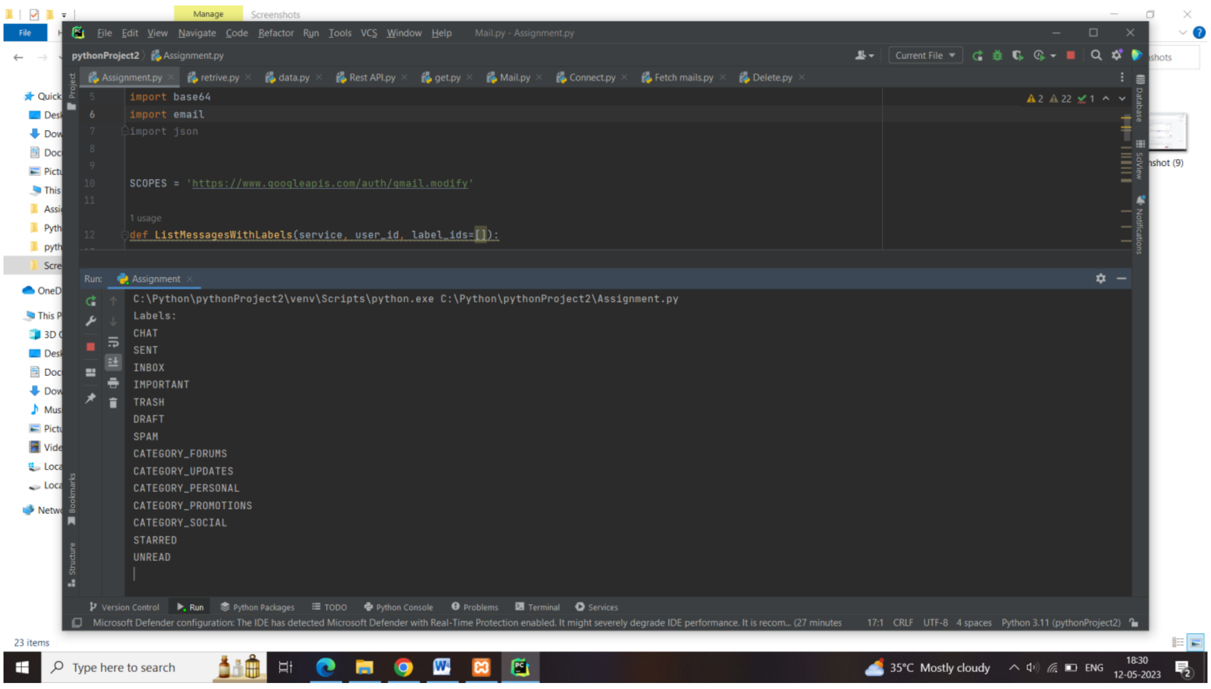


```
localhost:8080/?error=access_denied
The authentication flow has completed.
```

The authentication has completed.

STEP 3. LIST OF MAILS:

In the next step, I have a list of emails from my Gmail inbox, I need to set up a relational database system to store these emails. Next, Select a database system (such as SQL) and create a table to store my emails. Make sure to include all relevant data from the emails (such as the sender, subject, and time stamp).

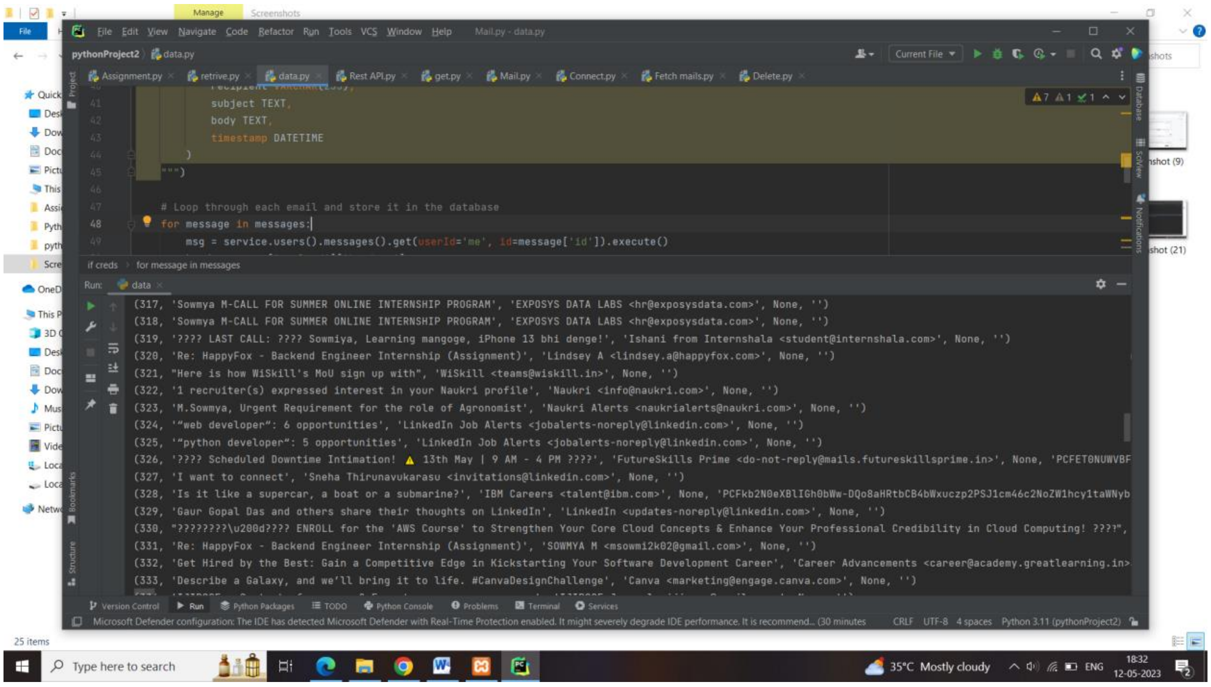


```python
import base64
import email
import json


SCOPES = 'https://www.googleapis.com/auth/gmail.modify'

def ListMessagesWithLabels(service, user_id, label_ids=[]):
```

```
C:\Python\pythonProject2\venv\Scripts\python.exe C:\Python\pythonProject2\Assignment.py
Labels:
CHAT
SENT
INBOX
IMPORTANT
TRASH
DRAFT
SPAM
CATEGORY_FORUMS
CATEGORY_UPDATES
CATEGORY_PERSONAL
CATEGORY_PROMOTIONS
CATEGORY_SOCIAL
STARRED
UNREAD
```

STEP 4. STORING EMAIL IN THE DATABASE :

To store emails in my database, use SQL queries to insert, update, or delete emails from the database table. Ensure that all relevant data from the emails are stored in the database along with the emails themselves.



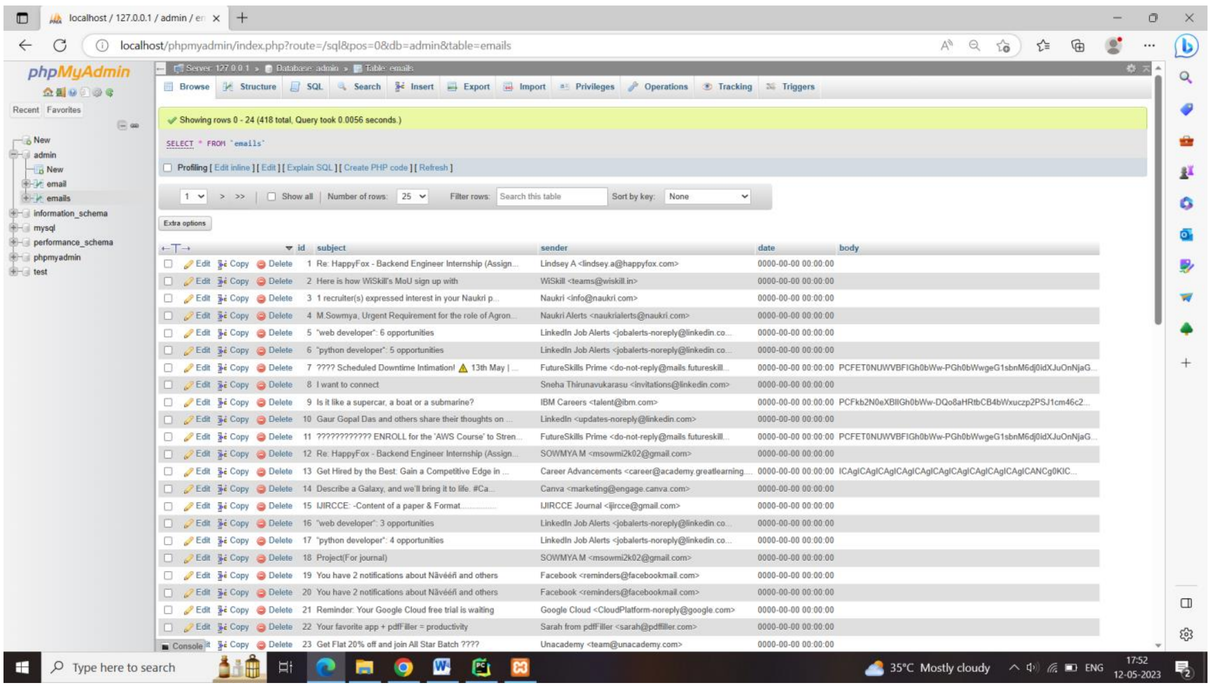
```
        subject TEXT,
        body TEXT,
        timestamp DATETIME
    )
""")

# Loop through each email and store it in the database
for message in messages:
    msg = service.users().messages().get(userId='me', id=message['id']).execute()
```

```
(317, 'Sowmya M-CALL FOR SUMMER ONLINE INTERNSHIP PROGRAM', 'EXPOSYS DATA LABS <hr@exposysdata.com>', None, '')
(318, 'Sowmya M-CALL FOR SUMMER ONLINE INTERNSHIP PROGRAM', 'EXPOSYS DATA LABS <hr@exposysdata.com>', None, '')
(319, '???? LAST CALL: ???? Sowmiya, Learning mangoge, iPhone 13 bhi denge!', 'Ishani from Internshala <student@internshala.com>', None, '')
(320, 'Re: HappyFox - Backend Engineer Internship (Assignment)', 'Lindsey A <lindsey.a@happyfox.com>', None, '')
(321, "Here is how WiSkill's MoU sign up with", 'WiSkill <teams@wiskill.in>', None, '')
(322, '1 recruiter(s) expressed interest in your Naukri profile', 'Naukri <info@naukri.com>', None, '')
(323, 'M.Sowmya, Urgent Requirement for the role of Agronomist', 'Naukri Alerts <naukrialerts@naukri.com>', None, '')
(324, '“web developer”: 6 opportunities', 'LinkedIn Job Alerts <jobalerts-noreply@linkedin.com>', None, '')
(325, '“python developer”: 5 opportunities', 'LinkedIn Job Alerts <jobalerts-noreply@linkedin.com>', None, '')
(326, '???? Scheduled Downtime Intimation! ⚠ 13th May | 9 AM - 4 PM ????', 'FutureSkills Prime <do-not-reply@mails.futureskillsprime.in>', None, 'PCFET0NUWVBF
(327, 'I want to connect', 'Sneha Thirunavukarasu <invitations@linkedin.com>', None, '')
(328, 'Is it like a supercar, a boat or a submarine?', 'IBM Careers <talent@ibm.com>', None, 'PCFkb2N0eXBlIGh0bWw-DQo8aHRtbCB4bWxuczp2PSJ1cm46c2NoZW1hcy1taWNyb
(329, 'Gaur Gopal Das and others share their thoughts on LinkedIn', 'LinkedIn <updates-noreply@linkedin.com>', None, '')
(330, "????????\u200d???? ENROLL for the 'AWS Course' to Strengthen Your Core Cloud Concepts & Enhance Your Professional Credibility in Cloud Computing! ????",
(331, 'Re: HappyFox - Backend Engineer Internship (Assignment)', 'SOWMYA M <msowmi2k02@gmail.com>', None, '')
(332, 'Get Hired by the Best: Gain a Competitive Edge in Kickstarting Your Software Development Career', 'Career Advancements <career@academy.greatlearning.in>
(333, 'Describe a Galaxy, and we'll bring it to life. #CanvaDesignChallenge', 'Canva <marketing@engage.canva.com>', None, '')
```

Microsoft Defender configuration: The IDE has detected Microsoft Defender with Real-Time Protection enabled. It might severely degrade IDE performance. It is recommend... (30 minutes   CRLF   UTF-8   4 spaces   Python 3.11 (pythonProject2)

This code retrieve data from Gmail using a python script to connect MySQL

STEP:CONNECT MAILS IN SQL:



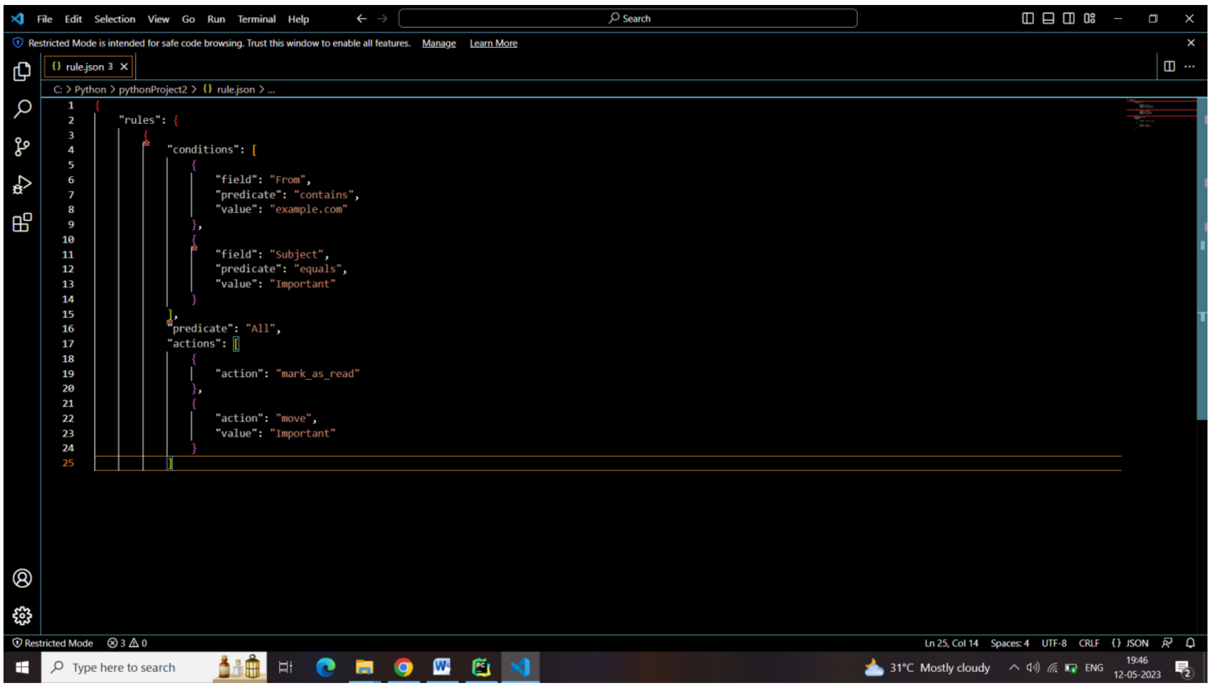
✓ Showing rows 0 - 24 (418 total, Query took 0.0056 seconds.)

```
SELECT * FROM `emails`
```

	id	subject	sender	date	body
Edit Copy Delete	1	Re: HappyFox - Backend Engineer Internship (Assign...	Lindsey A <lindsey.a@happyfox.com>	0000-00-00 00:00:00	
Edit Copy Delete	2	Here is how WiSkill's MoU sign up with	WiSkill <teams@wiskill.in>	0000-00-00 00:00:00	
Edit Copy Delete	3	1 recruiter(s) expressed interest in your Naukri p...	Naukri <info@naukri.com>	0000-00-00 00:00:00	
Edit Copy Delete	4	M.Sowmya, Urgent Requirement for the role of Agron...	Naukri Alerts <naukrialerts@naukri.com>	0000-00-00 00:00:00	
Edit Copy Delete	5	"web developer": 6 opportunities	LinkedIn Job Alerts <jobalerts-noreply@linkedin.co...	0000-00-00 00:00:00	
Edit Copy Delete	6	"python developer": 5 opportunities	LinkedIn Job Alerts <jobalerts-noreply@linkedin.co...	0000-00-00 00:00:00	
Edit Copy Delete	7	???? Scheduled Downtime Intimation! ⚠ 13th May | ...	FutureSkills Prime <do-not-reply@mails.futureskill...	0000-00-00 00:00:00	PCFET0NUWVBFIGh0bWw-PGh0bWwgeG1sbnM6dj0idXJuOnNjaG...
Edit Copy Delete	8	I want to connect	Sneha Thirunavukarasu <invitations@linkedin.com>	0000-00-00 00:00:00	
Edit Copy Delete	9	Is it like a supercar, a boat or a submarine?	IBM Careers <talent@ibm.com>	0000-00-00 00:00:00	PCFkb2N0eXBlIGh0bWw-DQo8aHRtbCB4bWxuczp2PSJ1cm46c2...
Edit Copy Delete	10	Gaur Gopal Das and others share their thoughts on ...	LinkedIn <updates-noreply@linkedin.com>	0000-00-00 00:00:00	
Edit Copy Delete	11	???????????? ENROLL for the 'AWS Course' to Stren...	FutureSkills Prime <do-not-reply@mails.futureskill...	0000-00-00 00:00:00	PCFET0NUWVBFIGh0bWw-PGh0bWwgeG1sbnM6dj0idXJuOnNjaG...
Edit Copy Delete	12	Re: HappyFox - Backend Engineer Internship (Assign...	SOWMYA M <msowmi2k02@gmail.com>	0000-00-00 00:00:00	
Edit Copy Delete	13	Get Hired by the Best: Gain a Competitive Edge in ...	Career Advancements <career@academy.greatlearning....	0000-00-00 00:00:00	ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICANCg0KIC...
Edit Copy Delete	14	Describe a Galaxy, and we'll bring it to life. #Ca...	Canva <marketing@engage.canva.com>	0000-00-00 00:00:00	
Edit Copy Delete	15	IJIRCCE: -Content of a paper & Format...............	IJIRCCE Journal <ijircce@gmail.com>	0000-00-00 00:00:00	
Edit Copy Delete	16	"web developer": 3 opportunities	LinkedIn Job Alerts <jobalerts-noreply@linkedin.co...	0000-00-00 00:00:00	
Edit Copy Delete	17	"python developer": 4 opportunities	LinkedIn Job Alerts <jobalerts-noreply@linkedin.co...	0000-00-00 00:00:00	
Edit Copy Delete	18	Project(For journal)	SOWMYA M <msowmi2k02@gmail.com>	0000-00-00 00:00:00	
Edit Copy Delete	19	You have 2 notifications about Nāvéėñ and others	Facebook <reminders@facebookmail.com>	0000-00-00 00:00:00	
Edit Copy Delete	20	You have 2 notifications about Nāvéėñ and others	Facebook <reminders@facebookmail.com>	0000-00-00 00:00:00	
Edit Copy Delete	21	Reminder: Your Google Cloud free trial is waiting	Google Cloud <CloudPlatform-noreply@google.com>	0000-00-00 00:00:00	
Edit Copy Delete	22	Your favorite app + pdfFiller = productivity	Sarah from pdfFiller <sarah@pdffiller.com>	0000-00-00 00:00:00	
Copy Delete	23	Get Flat 20% off and join All Star Batch ????	Unacademy <team@unacademy.com>	0000-00-00 00:00:00	

STEP 6. PROCESSING EMAILS:

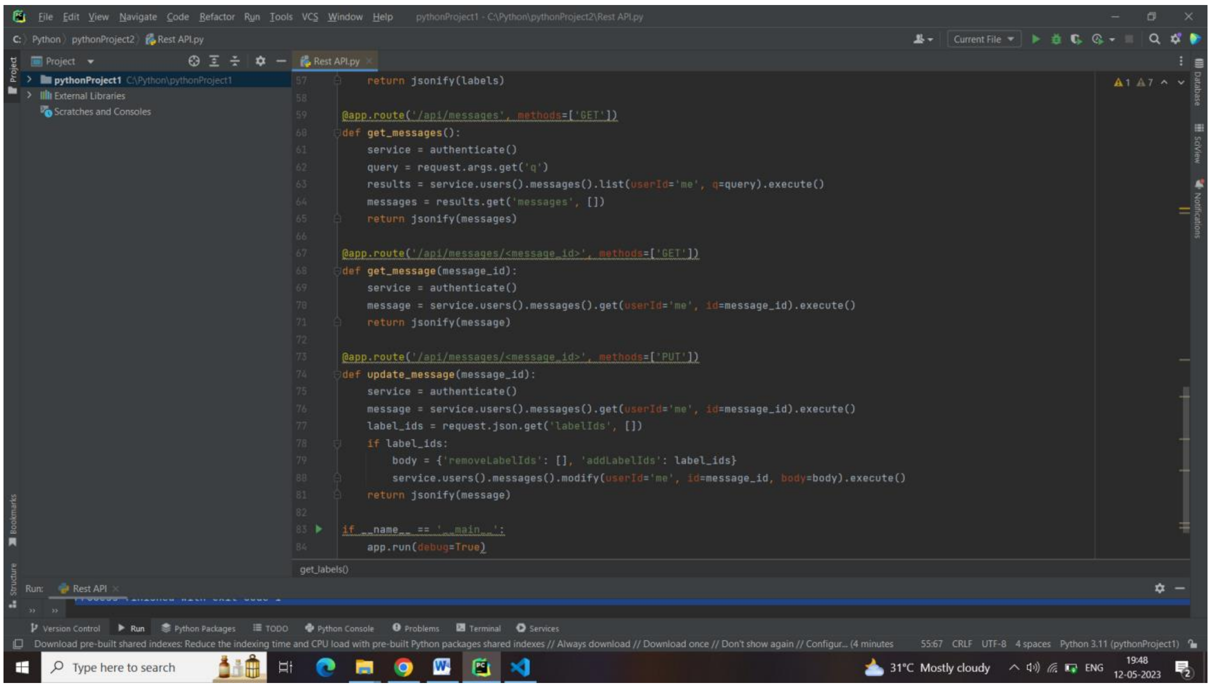
Process each email in the database based on the set of rules defined in the JSON file.



```
{
    "rules": {
        {
            "conditions": [
                {
                    "field": "From",
                    "predicate": "contains",
                    "value": "example.com"
                },
                {
                    "field": "Subject",
                    "predicate": "equals",
                    "value": "Important"
                }
            ],
            "predicate": "All",
            "actions": [
                {
                    "action": "mark_as_read"
                },
                {
                    "action": "move",
                    "value": "Important"
                }
            ]
```

STEP 7.IMPLEMENT REST API:

After processing emails using the rules engine, implement REST API endpoints that allow the user to take additional actions on the emails via an external interface. These might include endpoints for filtering emails by user-defined criteria, marking emails as read or unread, or sending emails to a different email client.



```
    return jsonify(labels)

@app.route('/api/messages', methods=['GET'])
def get_messages():
    service = authenticate()
    query = request.args.get('q')
    results = service.users().messages().list(userId='me', q=query).execute()
    messages = results.get('messages', [])
    return jsonify(messages)

@app.route('/api/messages/<message_id>', methods=['GET'])
def get_message(message_id):
    service = authenticate()
    message = service.users().messages().get(userId='me', id=message_id).execute()
    return jsonify(message)

@app.route('/api/messages/<message_id>', methods=['PUT'])
def update_message(message_id):
    service = authenticate()
    message = service.users().messages().get(userId='me', id=message_id).execute()
    label_ids = request.json.get('labelIds', [])
    if label_ids:
        body = {'removeLabelIds': [], 'addLabelIds': label_ids}
        service.users().messages().modify(userId='me', id=message_id, body=body).execute()
    return jsonify(message)

if __name__ == '__main__':
    app.run(debug=True)
```