Z

I have a Re folder including report.py, .env, games.pdf, how to send a daily report email to gracepan922@gmail.com from zpan01080124@gmail.com, using python way in vs code of doing this, make sure games.pdf is the attachment?

3/3

Sure, I can help you with that. You can use Python's built-in `smtplib` library along with `email.mime` to send an email with an attachment. Here's a basic script to achieve this:

```python
import smtplib from email.mime.multipart import MIMEMultipart from email.mime.text
import MIMEText from email.mime.base import MIMEBase from email import encoders import
os def send_email(sender_email, sender_password, receiver_email, subject, body,
attachment_path): # Create a multipart message message = MIMEMultipart() message["From"]
= sender_email message["To"] = receiver_email message["Subject"] = subject # Add body to
email message.attach(MIMEText(body, "plain")) # Open the file to be sent with
open(attachment_path, "rb") as attachment: # Add file as application/octet-stream #
Email client can usually download this automatically as attachment part =
MIMEBase("application", "octet-stream") part.set_payload(attachment.read()) # Encode
file in ASCII characters to send by email encoders.encode_base64(part) # Add header as
key/value pair to attachment part part.add_header( "Content-Disposition", f"attachment;
filename= {os.path.basename(attachment_path)}", ) # Add attachment to message and
convert message to string message.attach(part) text = message.as_string() # Log in to
email account server = smtplib.SMTP("smtp.gmail.com", 587) server.starttls()
server.login(sender_email, sender_password) # Send email server.sendmail(sender_email,
receiver_email, text) # Quit SMTP server server.quit() # Email details sender_email =
"zpan01080124@gmail.com" sender_password = "yourpassword" receiver_email =
"gracepan922@gmail.com" subject = "Daily Report" body = "Please find the daily report
attached." attachment_path = "Re/games.pdf" # Send email send_email(sender_email,
sender_password, receiver_email, subject, body, attachment_path)
```

Replace **`"yourpassword"`** with the actual password for the sender's email account. Ensure that you have enabled "less secure apps" access in your Gmail account settings, or use an app password if you have 2-factor authentication enabled.

Also, make sure that you have configured your Gmail account to allow "less secure apps" to access it. You can do this in your Google account settings.

Remember, it's not secure to hardcode your email password in your script. Consider using environment variables or a configuration file to store sensitive information. Additionally, it's a good practice to handle exceptions and errors gracefully in your script.

```
  File
"c:\Users\grace\VisualStudioProject\pythonPrograms\Re\report.py",
line 41, in send_email
    server.login(sender_email, sender_password)
  File
"C:\Users\grace\AppData\Local\Programs\Python\Python312\Lib\smt
plib.py", line 750, in login
    raise last_exception
    (code, resp) = self.auth(
                   ^^^^^^^^^^^
  File
"C:\Users\grace\AppData\Local\Programs\Python\Python312\Lib\smt
plib.py", line 662, in auth
    raise SMTPAuthenticationError(code, resp)
smtplib.SMTPAuthenticationError: (535, b'5.7.8 Username and
Password not accepted. For more information, go to\n5.7.8
https://support.google.com/mail/?p=BadCredentials 38308e7fff4ca-
2e95bf1a366sm26189111fa.126 - gsmtp')
PS C:\Users\grace\VisualStudioProject\pythonPrograms\Re> &
C:/Users/grace/AppData/Local/Programs/Python/Python312/python.e
xe c:/Users/grace/VisualStudioProject/pythonPrograms/Re/report.py
Traceback (most recent call last):
  File
"c:\Users\grace\VisualStudioProject\pythonPrograms\Re\report.py",
line 58, in <module>
    send_email(sender_email, sender_password, receiver_email, subject,
body, attachment_path)
  File
"c:\Users\grace\VisualStudioProject\pythonPrograms\Re\report.py",
line 41, in send_email
    server.login(sender_email, sender_password)
  File
"C:\Users\grace\AppData\Local\Programs\Python\Python312\Lib\smt
plib.py", line 750, in login
    raise last_exception
  File
"C:\Users\grace\AppData\Local\Programs\Python\Python312\Lib\smt
plib.py", line 739, in login
```

```
    (code, resp) = self.auth(
                   ^^^^^^^^^^
  File
"C:\Users\grace\AppData\Local\Programs\Python\Python312\Lib\smt
plib.py", line 662, in auth
    raise SMTPAuthenticationError(code, resp)
smtplib.SMTPAuthenticationError: (535, b'5.7.8 Username and
Password not accepted. For more information, go to\n5.7.8
https://support.google.com/mail/?p=BadCredentials 38308e7fff4ca-
2ea46e141acsm2699631fa.87 - gsmtp')   get stuck with the same
mistake agin, could it because I tried so many times?
```

The error you're encountering, `SMTPAuthenticationError: (535, b'5.7.8 Username and Password not accepted')`, typically occurs when the SMTP server rejects the login attempt due to incorrect credentials or if it considers the login attempt as suspicious (such as from an unfamiliar location or device).

Here are a few steps you can take to troubleshoot this issue:

1. **Double-check your credentials**: Ensure that you're using the correct email address and password for the sender's email account. Even a small typo can cause authentication failures.

2. **Less secure apps access**: Make sure that you've enabled the "Allow less secure apps" option in your Gmail account settings. This allows applications like your Python script to access your Gmail account using your password.

3. **Captcha requirement**: Sometimes, Google may require you to complete a CAPTCHA challenge if it detects suspicious activity. Try logging into your Gmail account through a web browser and see if you're prompted to complete any CAPTCHA challenges.

4. **App password (if using 2-factor authentication)**: If you have 2-factor authentication enabled on your Gmail account, you should generate an "app password" to use in your script instead of your regular account password. This app password is specifically generated for use with third-party applications and won't require you to disable 2-factor authentication or use your primary Google account password.

5. **IP-based restrictions**: Check if your Gmail account has any IP-based restrictions configured. If so, you may need to whitelist the IP address from which your script is attempting to connect.

6. **Recent activity**: If you've tried logging in multiple times with incorrect credentials, Google's security systems may temporarily block access. In such cases, wait for some time (usually a few hours) before attempting to log in again.

Try these steps and see if they resolve the issue. If you're still facing problems, let me know, and we can explore further options.

```
  File
"C:\Users\grace\AppData\Local\Programs\Python\Python312\Lib\smt
plib.py", line 739, in login
    (code, resp) = self.auth(
                   ^^^^^^^^^^
  File
"C:\Users\grace\AppData\Local\Programs\Python\Python312\Lib\smt
plib.py", line 662, in auth
    raise SMTPAuthenticationError(code, resp)
smtplib.SMTPAuthenticationError: (535, b'5.7.8 Username and
Password not accepted. For more information, go to\n5.7.8
https://support.google.com/mail/?p=BadCredentials 38308e7fff4ca-
2e95bdd14c7sm25472301fa.97 - gsmtp')
PS C:\Users\grace\VisualStudioProject\pythonPrograms\Re>   I have
2-factor authentication, but still not working
```

If you have 2-factor authentication enabled on your Google account, you should generate an "app password" to use instead of your regular account password. App passwords are specifically designed for use with applications that don't support 2-factor authentication.

Here's how you can generate an app password for your Python script:

1. Go to your Google Account settings page: https://myaccount.google.com/security
2. In the "Signing in to Google" section, select "App passwords." You may need to sign in again.
3. If prompted, re-enter your Google account password.
4. In the "App passwords" section, select "Select app" and choose "Other (Custom name)".
5. Enter a name for your app password, such as "Python Email Script," and click "Generate."
6. Google will generate a 16-character app password for you. Copy this password.

7. Use this app password in your Python script instead of your regular Google account password.

Replace the `sender_password` variable in your Python script with the app password you generated. Then try running the script again.

This should allow your Python script to authenticate with Google's SMTP server successfully, even with 2-factor authentication enabled. If you continue to encounter issues, double-check that you're using the correct app password and that it hasn't expired or been revoked.

```
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.application import MIMEApplication
from datetime import datetime
import os
from dotenv import load_dotenv

# Determine the directory of the current script
script_dir = os.path.dirname(os.path.abspath(__file__))

# Load environment variables from .env file located in the same directory as the script
load_dotenv(dotenv_path=os.path.join(script_dir, '.env'))

def send_email(subject, body, to_emails, attachment_paths=None):
    from_email = os.getenv('EMAIL_USER')
    password = os.getenv('EMAIL_PASSWORD')

    if not from_email or not password:
        raise ValueError("EMAIL_USER and EMAIL_PASSWORD environment variables must be set")

    # Create the email container
    msg = MIMEMultipart()
    msg['From'] = from_email
    msg['To'] = ', '.join(to_emails)
    msg['Subject'] = subject

    # Attach the email body
    msg.attach(MIMEText(body, 'plain'))

    # Attach files if specified
    if attachment_paths:
        for attachment_path in attachment_paths:
            with open(attachment_path, 'rb') as file:
                part = MIMEApplication(file.read(), Name=os.path.basename(attachment_path))
                part['Content-Disposition'] = f'attachment; filename="{os.path.basename(attachment_path)}"'
                msg.attach(part)

    # Send the email
    with smtplib.SMTP('smtp.gmail.com', 587) as server:
        server.starttls()
        server.login(from_email, password)
        server.sendmail(from_email, to_emails, msg.as_string())

def generate_report():
    # Replace with your report generation logic
    report_content = "Daily Report:\n\nAll systems operational."
    return report_content

if __name__ == '__main__':
    report = generate_report()
    subject = f"Daily Report - {datetime.now().strftime('%Y-%m-%d')}"
    body = report
    to_emails = ['gracepan922@gmail.com']

    # Add paths to attachments here
    attachment_paths = [
        os.path.join(script_dir, 'games.pdf')

    ]

    send_email(subject, body, to_emails, attachment_paths)
```

this is my code, what could be wrong

Your code seems generally well-structured for sending an email with attachments. However, there are a few potential issues that might be causing problems:

1. **Environment Variables:** Ensure that your `.env` file contains the correct values for `EMAIL_USER` and `EMAIL_PASSWORD`, and that it's located in the same directory as your script. You should replace `"your_email@gmail.com"` and `"your_password"` with your actual email address and password in the `.env` file.

2. **Less Secure Apps Access:** If you haven't already, make sure that "less secure apps" access is enabled for your Gmail account. You can do this by going to your Google Account settings and enabling it.
3. **App Password:** If you have 2-factor authentication enabled for your Gmail account, you need to generate an app password and use it in your script instead of your actual password.
4. **File Paths:** Double-check that the file paths provided in `attachment_paths` are correct and that the files exist in those locations. Ensure that `'games.pdf'` exists in the same directory as your script.
5. **SMTP Server Connection:** Ensure that your computer can connect to the Gmail SMTP server on port 587. Sometimes firewalls or antivirus programs can block such connections.
6. **Email Address Format:** Make sure that the email addresses provided in the `to_emails` list are valid and correctly formatted.
7. **Error Handling:** Implement error handling to catch any exceptions that might occur during the email sending process. This will help you identify the specific cause of the problem.

If you've checked all of these points and are still experiencing issues, try running the script in a different environment or with different credentials to see if the problem persists. Additionally, check the error messages or logs for more detailed information about what might be going wrong.

comment the place I should change

Sure, I'll comment the places where you need to make changes:

```python
import smtplib from email.mime.multipart import MIMEMultipart from email.mime.text
import MIMEText from email.mime.application import MIMEApplication from datetime import
datetime import os from dotenv import load_dotenv # Determine the directory of the
current script script_dir = os.path.dirname(os.path.abspath(__file__)) # Load
environment variables from .env file located in the same directory as the script
load_dotenv(dotenv_path=os.path.join(script_dir, '.env')) def send_email(subject, body,
to_emails, attachment_paths=None): from_email = os.getenv('EMAIL_USER') # Change
'EMAIL_USER' to the name of your environment variable containing your email password =
os.getenv('EMAIL_PASSWORD') # Change 'EMAIL_PASSWORD' to the name of your environment
```

```
variable containing your email password if not from_email or not password: raise
ValueError("EMAIL_USER and EMAIL_PASSWORD environment variables must be set") # Create
the email container msg = MIMEMultipart() msg['From'] = from_email msg['To'] = ',
'.join(to_emails) msg['Subject'] = subject # Attach the email body
msg.attach(MIMEText(body, 'plain')) # Attach files if specified if attachment_paths: for
attachment_path in attachment_paths: with open(attachment_path, 'rb') as file: part =
MIMEApplication(file.read(), Name=os.path.basename(attachment_path)) part['Content-
Disposition'] = f'attachment; filename="{os.path.basename(attachment_path)}"'
msg.attach(part) # Send the email with smtplib.SMTP('smtp.gmail.com', 587) as server:
server.starttls() server.login(from_email, password) server.sendmail(from_email,
to_emails, msg.as_string()) def generate_report(): # Replace with your report generation
logic report_content = "Daily Report:\n\nAll systems operational." return report_content
if __name__ == '__main__': report = generate_report() subject = f"Daily Report -
{datetime.now().strftime('%Y-%m-%d')}" body = report to_emails =
['gracepan922@gmail.com'] # Change this to the recipient email addresses # Add paths to
attachments here attachment_paths = [ os.path.join(script_dir, 'games.pdf') ] # Ensure
the file paths are correct send_email(subject, body, to_emails, attachment_paths) # Run
the function to send the email
```

Make sure to replace **`'EMAIL_USER'`** and **`'EMAIL_PASSWORD'`** with the actual names of the environment variables containing your email address and password. Also, update **`to_emails`** with the correct recipient email address(es), and verify that the **`attachment_paths`** list contains the correct paths to your attachments.

can you give me an easier way to conduct this I get stuck with the error althought all settings are correct?

If you're encountering difficulties with the SMTP approach, you might consider using an external library like **`yagmail`**, which simplifies the process of sending emails through Gmail in Python. It handles authentication and other complexities for you, making it easier to focus on your email content.

Here's how you can use **`yagmail`** to send an email with attachments:

First, you need to install **`yagmail`** if you haven't already:

```bash
pip install yagmail
```

Then, you can use the following code to send an email with attachments:

```python
import yagmail def send_email(subject, body, to_email, attachment_paths=None): #
Initialize yagmail SMTP connection yag = yagmail.SMTP('your_email@gmail.com',
'your_password') # Send email yag.send(to=to_email, subject=subject, contents=body,
attachments=attachment_paths) if __name__ == '__main__': subject = "Daily Report" body =
"Please find the daily report attached." to_email = "gracepan922@gmail.com" # Replace
with recipient email address attachment_paths = ["Re/games.pdf"] # Replace with correct
attachment path send_email(subject, body, to_email, attachment_paths)
```

Replace `'your_email@gmail.com'` and `'your_password'` with your Gmail email address and password. Replace `"gracepan922@gmail.com"` with the recipient email address, and `"Re/games.pdf"` with the correct path to your attachment.

`yagmail` should handle the rest for you, including authentication and attachment handling. If you encounter any issues, it's likely to be related to your Gmail account settings or network configuration rather than the code itself.

```
import yagmail

def send_email(subject, body, to_email, attachment_paths=None):
    # Initialize yagmail SMTP connection
    yag = yagmail.SMTP('zpan01080124@gmail.com',
'jonmhkxxddvnlquj')

    # Send email
    yag.send(to=to_email, subject=subject, contents=body,
attachments=attachment_paths)

if __name__ == '__main__':
    subject = "Daily Report"
    body = "Please find the daily report attached."
    to_email = "gracepan922@gmail.com"  # Replace with recipient
email address

    attachment_paths = ["games.pdf"]  # Replace with correct
attachment path

    send_email(subject, body, to_email, attachment_paths)
```

explain all these steps to me in detail, thanks

Sure, let's break down each step in detail:

1. **Importing `yagmail`**:

```python
import yagmail
```

Here, we're importing the `yagmail` library, which provides a simplified interface for sending emails through Gmail in Python.

2. **Defining the `send_email` Function**:

```python
def send_email(subject, body, to_email, attachment_paths=None): # Initialize yagmail
SMTP connection yag = yagmail.SMTP('zpan01080124@gmail.com', 'jonmhkxxddvnlquj') # Send
email yag.send(to=to_email, subject=subject, contents=body,
attachments=attachment_paths)
```

In this function:

- `subject`: This parameter represents the subject line of the email.
- `body`: This parameter contains the body text of the email.
- `to_email`: This parameter specifies the recipient email address.
- `attachment_paths`: This parameter is optional and contains a list of file paths for any attachments to be included in the email.

Inside the function:

- We initialize a `yagmail.SMTP` object, which establishes a connection to the SMTP server using the provided Gmail credentials (`'zpan01080124@gmail.com'` and `'jonmhkxxddvnlquj'`). This line of code essentially logs in to your Gmail account.
- Then, we use the `yag.send()` method to send the email. We specify the recipient (`to`), subject (`subject`), body (`contents`), and any attachments (`attachments`).

3. **Main Part of the Script**:

```python
if __name__ == '__main__': subject = "Daily Report" body = "Please find the daily
report attached." to_email = "gracepan922@gmail.com" # Replace with recipient email
address attachment_paths = ["games.pdf"] # Replace with correct attachment path
send_email(subject, body, to_email, attachment_paths)
```

Here, we're executing the **`send_email`** function with some sample data.

- We define the **`subject`**, **`body`**, **`to_email`**, and **`attachment_paths`** variables.
- Then, we call the **`send_email`** function with these variables as arguments, triggering the email sending process.

Remember to replace **`'zpan01080124@gmail.com'`** and **`'jonmhkxxddvnlquj'`** with your actual Gmail email address and password, respectively. Also, replace **`"gracepan922@gmail.com"`** with the email address of the recipient, and **`"games.pdf"`** with the correct path to your attachment file.

After running this script, an email with the specified subject, body, and attachment will be sent to the specified recipient.