# Python Code

## Send Pricing

You've made an excellent choice asking about the script solution. After watching you battle with Power Automate's limitations, I can tell you that a simple Python script will give you exactly what you need with far less frustration. Let me walk you through why this approach works so well and how to implement it.

**Why a Script Succeeds Where Power Automate Failed**

Think of Power Automate as trying to communicate through a broken telephone - it keeps garbling your message between systems. A Python script, on the other hand, speaks directly to Outlook using Microsoft's official language (their API). It's like the difference between playing telephone and having a direct conversation.

When you use Python with the Outlook API, you're telling Outlook exactly what you want: "Create a draft email, format it as HTML, put it in the drafts folder." There's no translation layer adding unwanted HTML tags or misinterpreting your intentions.

**Understanding the Basic Approach**

The script would work in three simple steps:

1. Read your customer data from an Excel file (customer names, emails, file paths)
2. For each customer, create a properly formatted HTML email draft
3. Attach the appropriate pricing file to each draft

Let me show you what this actually looks like in practice:**Understanding What Makes This Script Special**

Looking at the code above, you can see how straightforward this approach is compared to wrestling with Power Automate. The script does exactly what you've been trying to achieve: it creates properly formatted HTML emails as drafts in Outlook, with your company colors, bold text, and professional formatting all intact.

The key line that makes everything work is mail.HTMLBody = .... This tells Outlook explicitly "this is HTML content, please render it as such." There's no ambiguity, no translation errors, no mysterious paragraph tags appearing where they shouldn't. It's like speaking directly to Outlook in its native language instead of through a faulty translator.

**What You'll Need to Set This Up**

Think of this setup process like assembling a simple toolkit. You'll need three main components:

1. **Python**: This is the programming language (free from python.org)
2. **A few helper libraries**: These are like specialized tools that help Python talk to Outlook and read Excel files
3. **Your existing Outlook**: The script works with the Outlook already on your computer

Here's the simple setup process:

1. **Install Python** (if you don't have it already):
   * Go to python.org and download Python 3.9 or newer
   * During installation, make sure to check "Add Python to PATH"
2. **Install the required libraries** by opening Command Prompt and typing:
3. pip install pywin32 pandas openpyxl
4. **Prepare your Excel file** with these columns:
   * CustomerName (like "Atlantic Performance Oils")
   * RecipientName (like "Arnie, Steve, Pares, and Brad")
   * EmailAddresses (like "arnie@atlantic.com; steve@atlantic.com")
   * FileName (like "250601\_Pricing\_Atlantic Performance Oils.pdf")
   * FilePath (full path like "C:\Pricing\250601\_Pricing\_Atlantic Performance Oils.pdf")

**How to Use the Script**

Once everything is set up, using the script is remarkably simple:

1. Save the Python script to a file (like create\_drafts.py)
2. Update the excel\_file path in the script to point to your Excel file
3. Double-click the script file or run it from Command Prompt

The script will then create all your drafts, showing progress as it goes. You'll see messages like:

* "✓ Created draft for Atlantic Performance Oils"
* "✓ Created draft for XYZ Chemicals"

When it's done, all your drafts will be sitting in Outlook's Drafts folder, properly formatted with:

* Mark Anderson's name in bold
* Valorem Chemicals in your company blue color
* Proper paragraph spacing
* The disclaimer in smaller text
* Attached pricing PDFs

**Understanding Why This Succeeds**

The fundamental difference here is control. With Power Automate, you were at the mercy of how it chose to interpret and translate your content. With this script, you're giving direct, unambiguous instructions to Outlook. It's like the difference between asking someone to paint your house through a game of charades versus simply showing them the color swatches and pointing where to paint.

**Customization Options**

The beautiful thing about this approach is how easily you can customize it. Want to add different signatures for different team members? Simple. Need to customize the email body based on customer type? Just add an if-statement. Want to log which drafts were created? Add a few lines to write to a log file.

This flexibility means that as your needs evolve, your solution can evolve with them. No more fighting with rigid tools that don't quite do what you need.

**Your Next Steps**

Would you like me to walk you through the setup process in more detail? Or perhaps you'd like to see how to modify the script for your specific needs? For instance, if your Excel file has different column names or you want to add more sophisticated logic for different customer types, I can show you exactly how to adjust the code.

The relief you'll feel when you see those properly formatted drafts appear in Outlook - with all your formatting intact and no HTML tags in sight - will make the small setup effort worthwhile. No more fighting with Power Automate's limitations. Just clean, professional emails ready for your review.

import win32com.client

import pandas as pd

import os

from datetime import datetime

def create\_email\_drafts():

"""

Creates draft emails in Outlook with proper HTML formatting and attachments.

This script reads customer data from Excel and creates a draft for each customer.

"""

# Step 1: Connect to Outlook

# This creates a connection to your local Outlook application

print("Connecting to Outlook...")

outlook = win32com.client.Dispatch("Outlook.Application")

# Step 2: Read customer data from Excel

# Adjust this path to where your Excel file is located

excel\_file = r"C:\Users\MarkAnderson\Valorem\Knowledge Hub - Documents\Pricing\Customer Price Lists\Price Sheet Sending\_Python\Python\_CustomerPricing.xlsx"

print(f"Reading customer data from {excel\_file}")

# Read the Excel file - the headers are on row 3 (0-indexed)

df = pd.read\_excel(excel\_file, header=3)

# Clean column names (remove any trailing spaces)

df.columns = df.columns.str.strip()

# Show what columns we found

print(f"Found columns: {', '.join(df.columns)}")

print(f"Found {len(df)} customer records\n")

# Step 3: Create a draft for each customer

drafts\_created = 0

for index, row in df.iterrows():

try:

# Create a new email draft

mail = outlook.CreateItem(0) # 0 = Mail item

# Set the recipients

mail.To = row['EmailAddresses']

# Set CC and BCC if needed

mail.CC = "support@valorem.com.au;jasonn@valorem.com.au"

mail.BCC = "tonyb@valorem.com.au"

# Set the subject

mail.Subject = f"Monthly Pricing Update for {row['CustomerName']}"

# Create the HTML body with proper formatting

mail.HTMLBody = f"""

<html>

<body style="font-family: Arial, sans-serif;">

<p>Hi {row['RecipientName']},</p>

<p>Just sharing the latest pricing details for your account – the file's attached.</p>

<p>If you have any questions or would like to go over anything in more detail,

feel free to get in touch.</p>

<p>Thanks,</p>

<p>

<strong>Mark Anderson</strong><br>

Managing Director<br>

<strong style="color: rgb(74, 144, 226);">Valorem Chemicals Pty Ltd</strong><br>

Phone: +61 417 725 006<br>

Email: marka@valorem.com.au<br>

Web: www.valorem.com.au

</p>

<p style="font-size: 10px;">

This email and any files transmitted with it are confidential and

intended solely for the use of the individual or entity to whom they are addressed.

</p>

</body>

</html>

"""

# Attach the local file (FilePath = folder, FileName = filename)

folder = row.get('FilePath', '').strip()

filename = row.get('FileName', '').strip()

if folder and filename:

fullpath = os.path.join(folder, filename)

if os.path.exists(fullpath):

mail.Attachments.Add(fullpath)

else:

print(f"⚠ File not found for {row['CustomerName']}: {fullpath}")

# Save as draft (not send)

mail.Save()

print(f"✓ Created draft for {row['CustomerName']} ({row['EmailAddresses']})")

drafts\_created += 1

except Exception as e:

print(f"✗ Error creating draft for {row['CustomerName']}: {str(e)}")

print(f"\nCompleted! Created {drafts\_created} draft emails.")

print("Check your Outlook Drafts folder to review before sending.")

# Run the script

if \_\_name\_\_ == "\_\_main\_\_":

create\_email\_drafts()