







Approach

Build a script, go through gmail pull out the first and last name of client

- Retrieve email contents
- Scan the texts and fetch only clients' names from the email signature (case insensitive). Also look at the subject line, process email if it contains a new user.
- Map the name to the email address
- If multiple emails contain the same email address but with different signatures (typos or switching between nickname and official name). Store all of them in a list and associate that list to the email address.
- Create two different email labels Success & Failure, one containing all emails that were successfully fetched, while the other storing email that didn't work due to missing email signature or other reasons
- Stored the data in local mongoDB database
- Automate whenever a new email receives

To Do

- Connect to Gmail API, and fetch email from specific labels 
- Retrieve email with subject line containing keywords "New User(s)" 
- Ability to add label tags to email 
- Retrieve the name from email contents 
- Connect to local MongoDB 
- Automate the script 

Process using Python

Current Method: Use OAuth 2.0 to access Gmail API

- Complete the OAuth Consent Screen

<https://console.cloud.google.com/apis/credentials?authuser=1&project=python-email-script-421512&supportedpurview=project>

- In Credentials tab, go to Create OAuth ID Credential of type **Desktop**, then download the json file and add it to the working directory
- Install Google client library using

Python

```
py -m pip install --upgrade google-api-python-client  
google-auth-httpplib2 google-auth-oauthlib
```

- When executing for the first time, go to Enabled APIs & services to authorize access
- Recommend website to test out regex: <https://regex101.com/>
- Regex expression to capture name from email signature (subject to change)

Unset

```
(Best Regard(s)?|Sincerely|Thank you|Thank(s)*), (  
)*((\r\n|\r|\n))+([a-zA-Z][a-zA-z @\d]*)
```

- To connect the script to MongoDB, install the following module

Unset

```
py -m pip install pymongo
```

- Some method to automate the script:
<https://www.redwood.com/article/python-job-scheduling/>
- Use schedule module to configure when to call Gmail API

Unset

```
py -m pip install schedule
```

- Use Task Scheduler to set when to execute the script:
<https://www.youtube.com/watch?v=4n2fC97MNac&t=284s>

Previous Method (Doesn't work since company's account don't support third party app password)

- Python has a module called imaplib that can connect to the gmail account which can then read and post email
- The gmail account needs to enable IMAP access (gmail → settings → Forward and POP/IMAP → IMAP access: Enabled IMAP)
- Enable 2FA for the account
- Generate app password which the python requires to access