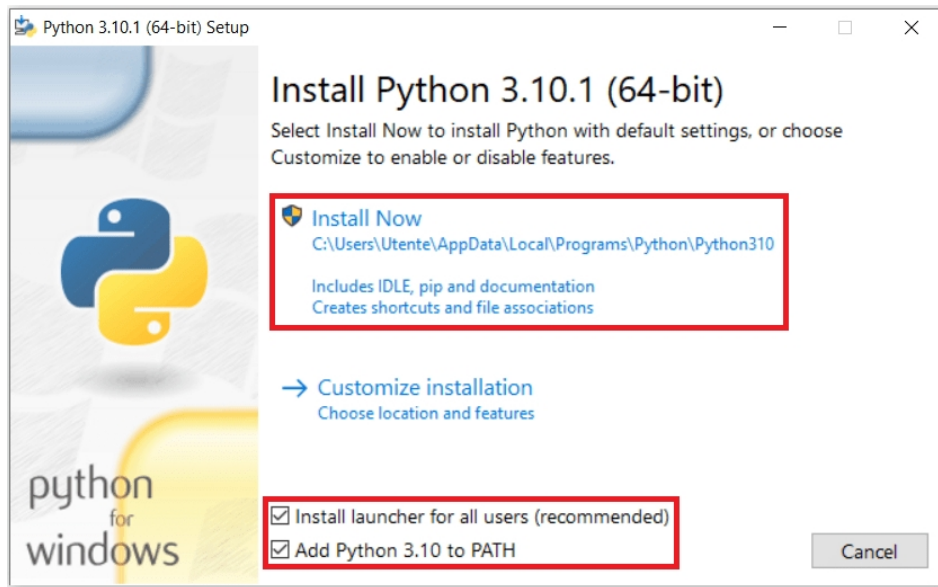




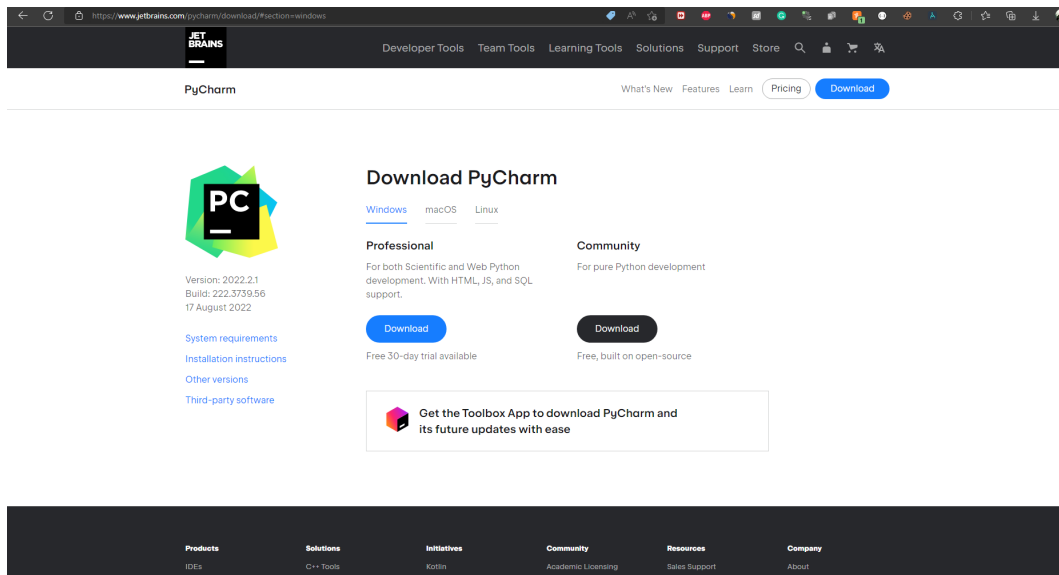
Project: Gotta act Professional with my Emails
AKA: Free Python version of MailChimp
AKA: email sender

1 Installing

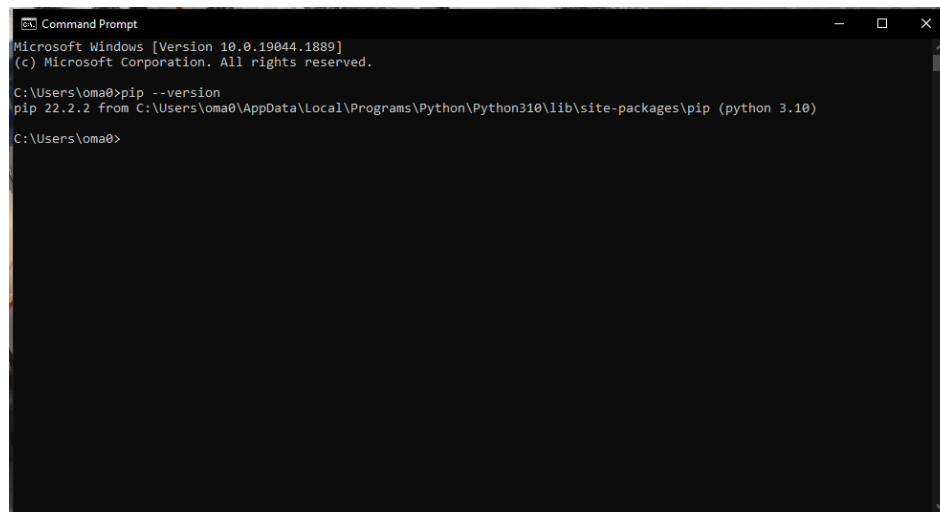
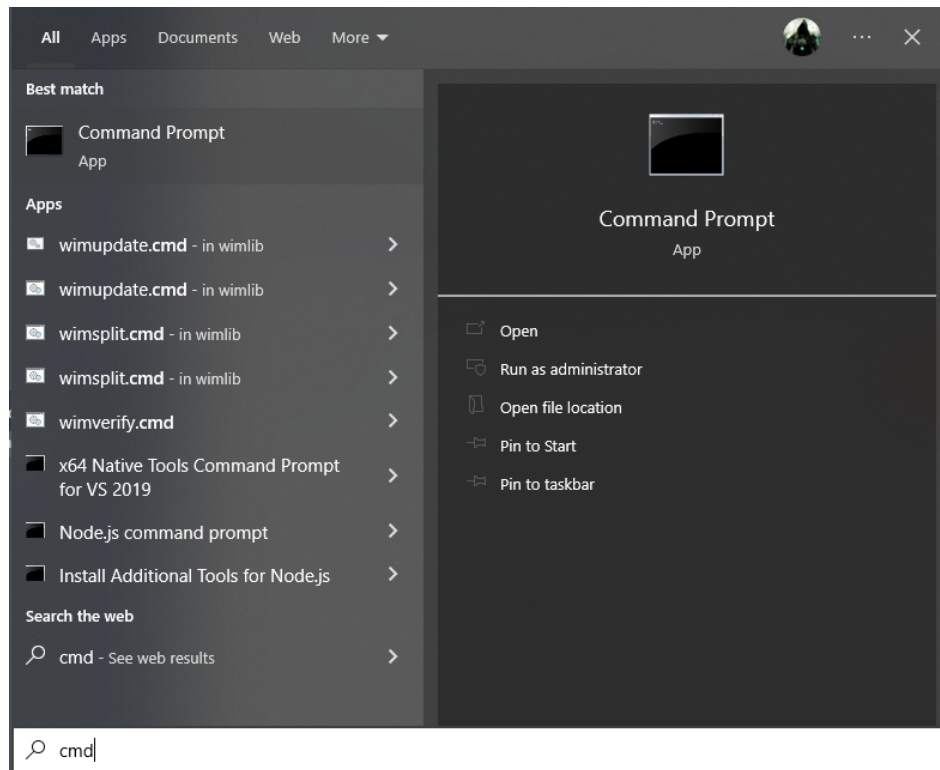
first of all, you need Python, you can download it from [here](#)
make sure before you start installing you add python to PATH



next up you need an Editor, you have a lot of options, for me, I use [PyCharm](#).

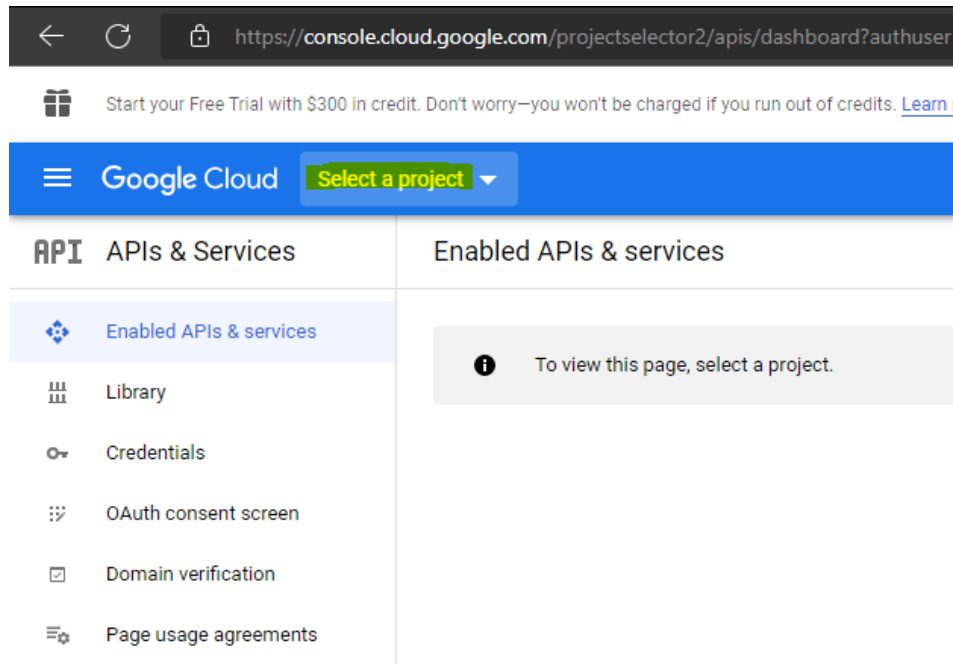


after installing Python, you can check if it went well by typing **pip** in **cmd**

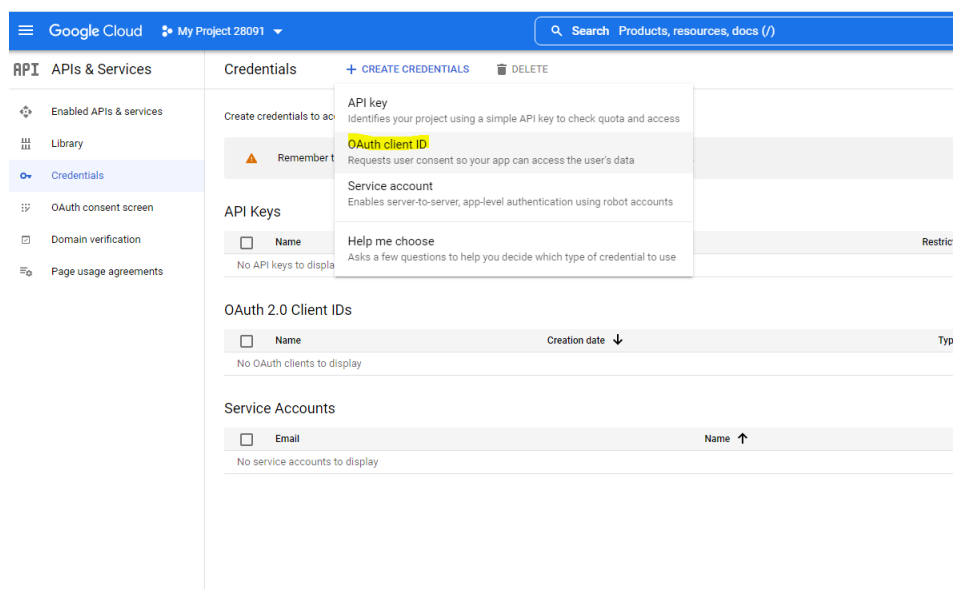


since we only need one package for the Email sender to work, since we already have cmd opened and pip in working might as well install it. type **pip install yagmail** in **cmd** to install the package

Since google security is 2nd to none you cant simply add your email and start sending, you need first is to create an app in google developers site to allow the code to send emails by using this app. first go here [google developers](#) after you logged in Select or create a project.



after you have selected your project go to credentials and create a new one, since you only created a project and you don't have any apps in it, it will redirect you to create one, just follow along.



just fill in the necessary* info or if you are feeling like it, fill everything.

Start your Free Trial with \$300 in credit. Don't worry—you won't be charged if you run out of credits. [Learn more](#)

Google Cloud My Project 28091

Search Products, resources, docs

APIs & Services

Enabled APIs & services

Library

Credentials

OAuth consent screen

Domain verification

Page usage agreements

Edit app registration

1 OAuth consent screen — 2 Scopes — 3 Test users — 4 Summary

App information

This shows in the consent screen, and helps end users know who you are and contact you

App name *

Email Sebder

The name of the app asking for consent

User support email *

testing.email.git@gmail.com

For users to contact you with questions about their consent

App logo

BROWSE

Upload an image, not larger than 1MB on the consent screen that will help users recognize your app. Allowed image formats are JPG, PNG, and BMP. Logos should be square and 120px by 120px for the best results.

App domain

To protect you and your users, Google only allows apps using OAuth to use Authorized Domains. The following information will be shown to your users on the consent screen.

Application home page

Provide users a link to your home page

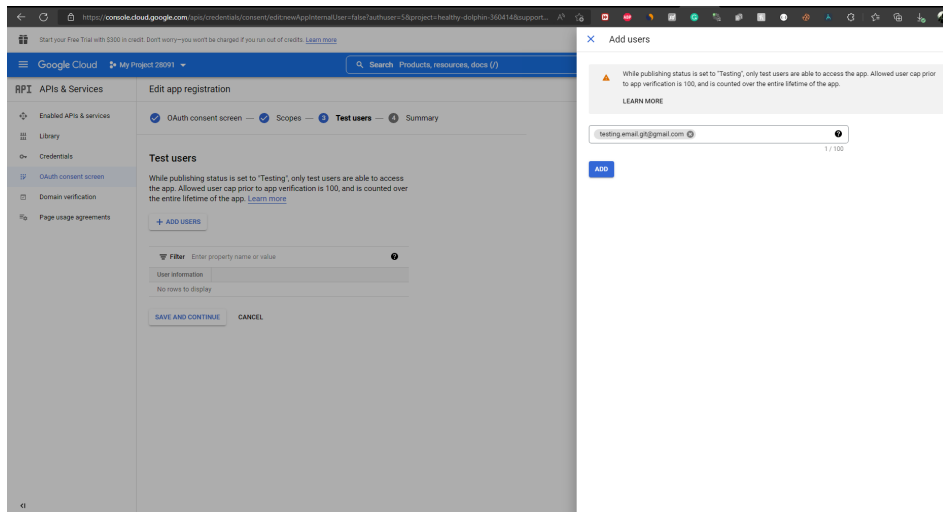
Application privacy policy link

Provide users a link to your public privacy policy

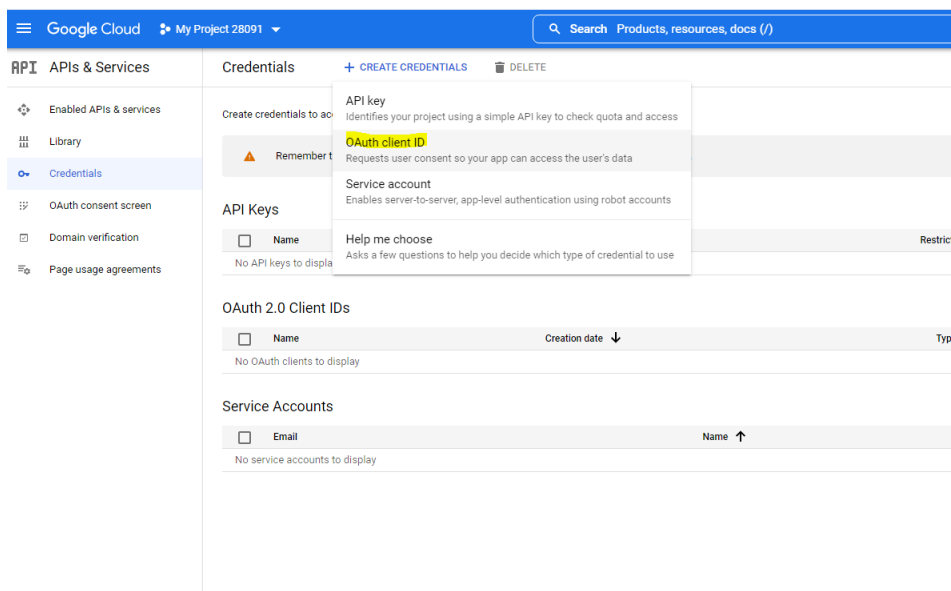
Application terms of service link

Provide users a link to your public terms of service

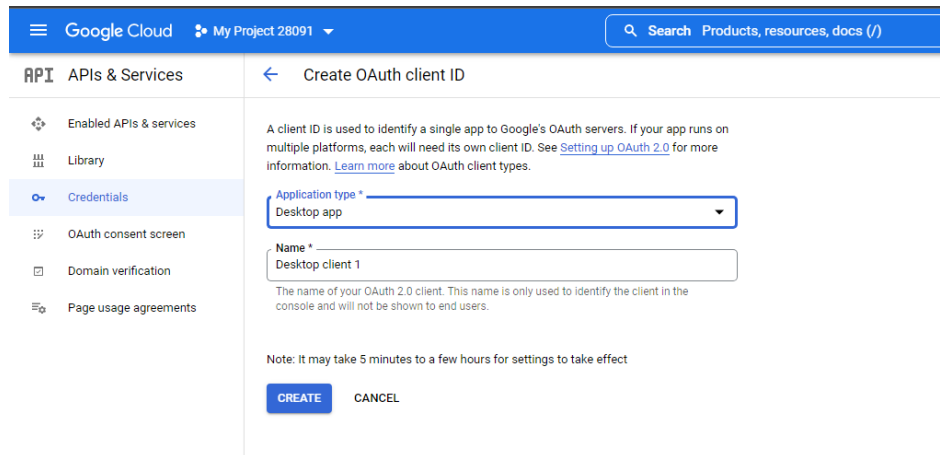
when you reach step 3, make sure you add test users, this user is the one who will be using this app, in other way this user is the one who will sending the emails. if you dont add anyone, the app will not work as there is no one is allowed to use it.



after that go back to the credentials and repeat the same step from before



since we are using python code, it counts as a Desktop app, select that and give it a name, or dont, it will automatically generate a name for it.



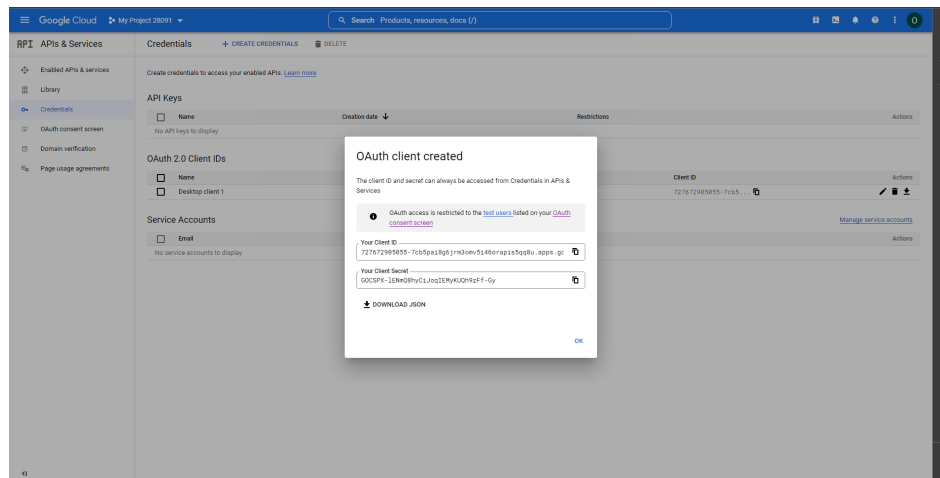
The screenshot shows the 'Create OAuth client ID' page in the Google Cloud console. The left sidebar has 'APIs & Services' selected, with 'Credentials' highlighted. The main content area has a heading 'Create OAuth client ID' and a subheading 'A client ID is used to identify a single app to Google's OAuth servers. If your app runs on multiple platforms, each will need its own client ID. See [Setting up OAuth 2.0](#) for more information. [Learn more](#) about OAuth client types.'

The form contains the following fields:

- Application type ***: A dropdown menu with 'Desktop app' selected.
- Name ***: A text input field containing 'Desktop client 1'. Below it is a note: 'The name of your OAuth 2.0 client. This name is only used to identify the client in the console and will not be shown to end users.'

At the bottom, there is a note: 'Note: It may take 5 minutes to a few hours for settings to take effect' and two buttons: 'CREATE' and 'CANCEL'.

The final step, download the OAuth JSON file, this is what we will be using to authenticate the process later on.



The screenshot shows the 'Credentials' page in the Google Cloud console. A modal dialog box titled 'OAuth client created' is open in the center. The dialog contains the following information:

- A message: 'The client ID and secret can always be accessed from Credentials in APIs & Services'.
- A note: 'OAuth access is restricted to the [test users](#) listed on your [OAuth consent screen](#)'.
- Your Client ID**: 727872985855-7c8f5a18c7jrm0xov5l460raps8qpfu.apps.googleusercontent.com
- Your Client Secret**: GOCSPx-32Med0hyCLJnQ2RyKjQh7cF-f0y
- A button: 'DOWNLOAD JSON'
- An 'OK' button at the bottom right.

The background shows the 'Credentials' page with sections for 'API Keys', 'OAuth 2.0 Client IDs', and 'Service Accounts'. The 'OAuth 2.0 Client IDs' section shows the newly created client with its ID and a 'Manage service accounts' link.

2 Setup

after all that, go to Git and clone or download the code here [GitCode](#)

there are 3 small changes that need to be made for the code to work correctly

first, change the **email** to your email address, and add the **OAuth2 JSON** we got in the same directory.

if you have an **attachment** you want to send with the emails also have it in the same file and change its name here i have an image but you can change that to anything

```
if __name__ == '__main__':  
  
    email_list = get_email_dict()  
    print_emails_dict(email_list)  
    img = 'image.jpg'  
  
    input("\nSend Emails? ")  
  
    yag = yagmail.SMTP("testing.email.git@gmail.com", oauth2_file="oauth2_creds.json")  
    for email, name in email_list.items():  
        subject, rec, body = get_email_body(name)  
        yag.send(to=email, subject=subject, contents=[body, img])  
        print(f"mail Sent To {email}")
```

there are two text files in the directory, one for emails, and the other for the email body, make sure the format for the emails is **Email:Name** since we are doing personalize emails this is a must.

```
ShadowFiend@gmail.com:NeverMore  
someone@live.com:Name  
Invoker@outlook.sa:Keal  
Osama@gmail.com:OSama
```

the body text field needs to be formatted like this:

the first time is the title, the second line needs to have only the greetings, since we have the names from the email's file the name will be added right next to the word 'Hello'

```
Subject Line: New [Lead magnet type]) [Lead magnet name]
Hello

I'm emailing you today to let you know we have created a new
In this [lead magnet type], you'll learn how to create [desc
[Link]Click here to access the [lead magnet name] ⇒[Link]

If you know anybody else who'll find this useful, please for
Let us know if you face any problems accessing the [lead mag
Thank you,
[Your signature]
```

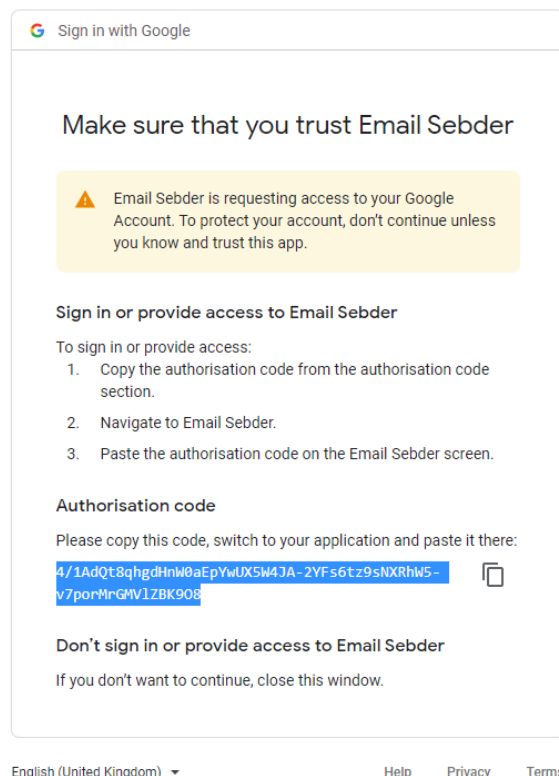
3 Testing the code

for the first run it will ask you to enter your email, (yes I know its there, but it will ask you to enter it again) and then prompt you to login to one user we added in the app previously

```
Email                                     Name
-----
ShadowFiend@gmail.com                   NeverMore
someone@live.com                       Name
Invoker@outlook.sa                     Keal
Osama@gmail.com                        OSama

Send Emails? yes
Your 'email address': testing_email_git@gmail.com
Navigate to the following URL to auth:
https://accounts.google.com/o/oauth2/auth?client\_id=727672905055-7cb5pai8q6jrm3o
Enter verification code: |
```

after you open the new link in the browsers, log in to the user, click continue, then allow, and then copy and paste the auth code in the IDE.



after that, you are done.