This program was developed in Python 3.8.3 and also tested in Python 3.7.6.

1) This project may be cloned from https://github.com/cyohman/darntough.git

2) I developed the API using Flask. I have never developed an API in Python before. All my previous Python experience was writing programs using the core API, GDAL, ogr, pandas, I matplotlib, and the Trade API. I looked at Flask and it seemed straightforward. Had I become stuck then I would have tried Django. My choices are usually based on speed to implement and working within the constraints of the environment I am in.

3) I used FlaskMail to send the email. Originally I tried using smtplib and debugging against command line local smtp server- python -m smtpd -n -c DebuggingServer localhost:1025 . I found using smtplib to be cumbersome and I could not accurately tell through the command line if my emails were being sent properly. Given issues that can arise from sending lots of email and being marked as a spammer, I opted to use: mailtrap.io. It's a solution to be able to send email to a SMTP server without sending it to its end destination, but still being able to look at the contents of the email. For the purposes of this demo, you can use my account to view the emails sent to the server.
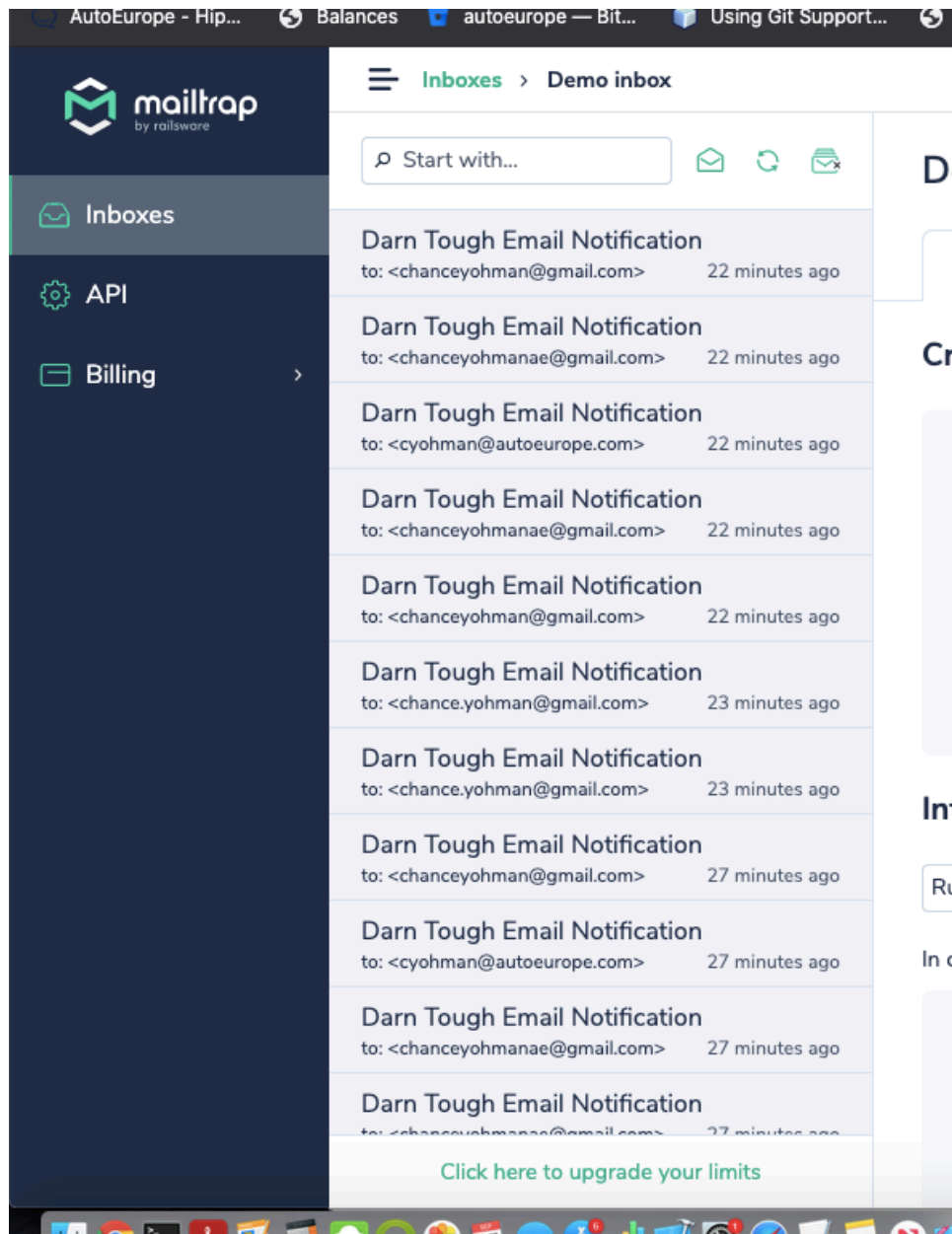
URL: mailtrap.io
Login: chance.yohman@gmail.com
Password: darntoughcodeprompt

Once logged on, click on Demo Inbox:
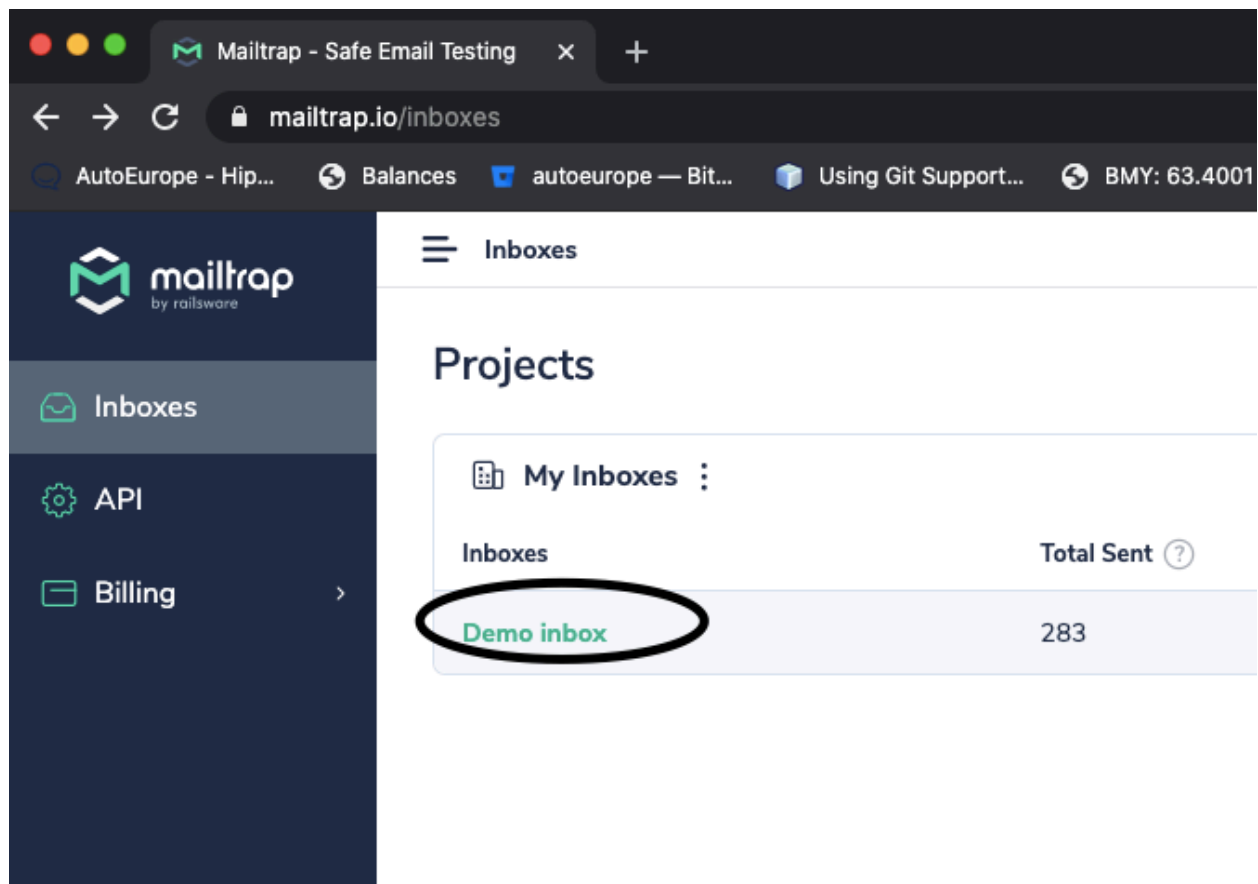
4) I named the endpoint- email. There is one required parameter- recipient. I could have made the subject, the from, and the body also parameters, but those did not seem to be the requirements for this project. I hardcoded the sender to be- noreply@darntough.com, the subject to be- Darn Tough Email Notification, and the body to be set based on whether or not there is an attachment. If no recipient is posted to the email then I return a 400 (Bad Request). An email can not go anywhere without a recipient.

Along those lines I also tried to do some checking on the endpoint to make sure the email was valid. I tried using email_validator, but abandoned that after running into dependency issues. I found a regex that I thought might work (always a risk), but found that email prefixes with two periods in the prefix fail the check. So email validation is incomplete in the endpoint.

I create the message with the subject "Darn Tough Email Notification" and set the recipients

to the recipient passed in. Next I check for the file parameter. If it's there, then I sent the body to "There is an attachment.". It's a good cue to go and check the attachment during

debugging to make sure the attachment came across. Finally I add the attachment. If there is no file parameter, then I set the body to "There is no attachment." and I make sure there is attachment when I check it in mailtrap.io.

Finally, I send the message. If there is no exception, then I return a 200 for a success. If there is an exception, then I return a 500 for Internal System Error.

5) I use virtual environment for running the application.

```
python3 -m venv venv

I am on a Mac (should work for *nix too), so after this I run:

source venv/bin/activate

On Windows, it should be:

venv\Scripts\activate.bat


To install any requirements, run:
```

```
[(base) cyohman@Chances-MacBook-Air darnTough % python3 -m venv venv
[(base) cyohman@Chances-MacBook-Air darnTough % source venv/bin/activate
[(venv) (base) cyohman@Chances-MacBook-Air darnTough % pip install -r requirements.txt
```

```
    I believe I aggregated all the requirements in requirements.txt. I
checked it on my MacBook Air and my iMac. If I missed something, then
let me know.

6) To run the test do: python -m pytest
```
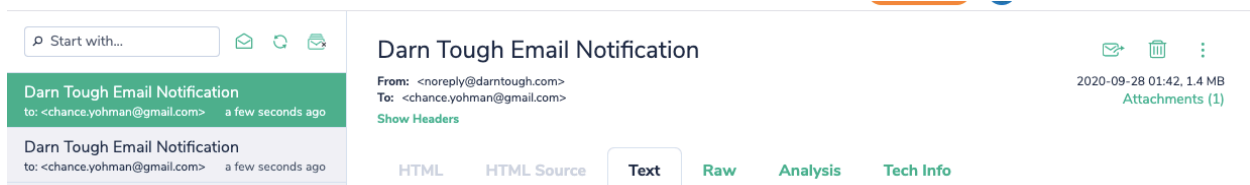
```
(venv) (base) cyohman@Chances-MacBook-Air darnTough % python -m pytest
=================================== test session starts ===================================
platform darwin -- Python 3.8.3, pytest-6.1.0, py-1.9.0, pluggy-0.13.1
rootdir: /Users/cyohman/Documents/darnTough
collected 6 items

tests/test_invalid_email.py .                                                        [ 16%]
tests/test_invalid_email_and_attachment.py .                                         [ 33%]
tests/test_no_email.py .                                                             [ 50%]
tests/test_no_email_and_attachment.py .                                             [ 66%]
tests/test_valid_email.py .                                                          [ 83%]
tests/test_valid_email_and_attachment.py .                                         [100%]

=================================== 6 passed in 5.85s =====================================
```
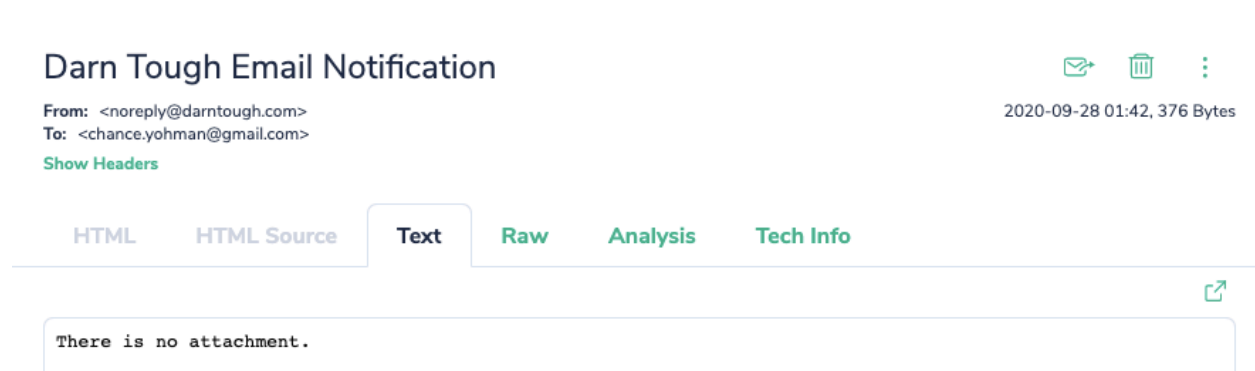
```
If you goto mailtrap.io, then you can see the emails come in for the
valid tests.
```

Start with...	Darn Tough Email Notification	2020-09-28 01:42, 1.4 MB
	From: <noreply@darntough.com>	Attachments (1)
Darn Tough Email Notification	To: <chance.yohman@gmail.com>	
to: <chance.yohman@gmail.com> a few seconds ago	Show Headers	
Darn Tough Email Notification		
to: <chance.yohman@gmail.com> a few seconds ago	HTML HTML Source **Text** Raw Analysis Tech Info	

Valid email with no attachment:

Darn Tough Email Notification

From: <noreply@darntough.com>
To: <chance.yohman@gmail.com>
Show Headers

2020-09-28 01:42, 376 Bytes

| HTML | HTML Source | Text | Raw | Analysis | Tech Info |

There is no attachment.

Valid email with an attachment:

Darn Tough Email Notification

From: <noreply@darntough.com>
To: <chance.yohman@gmail.com>
Show Headers

2020-09-28 01:42, 1.4 MB
Attachments (1)

ct_section13.pdf   1.1 MB

| HTML | HTML Source | Text | Raw | Analysis | Tech Info |

There is an attachment.
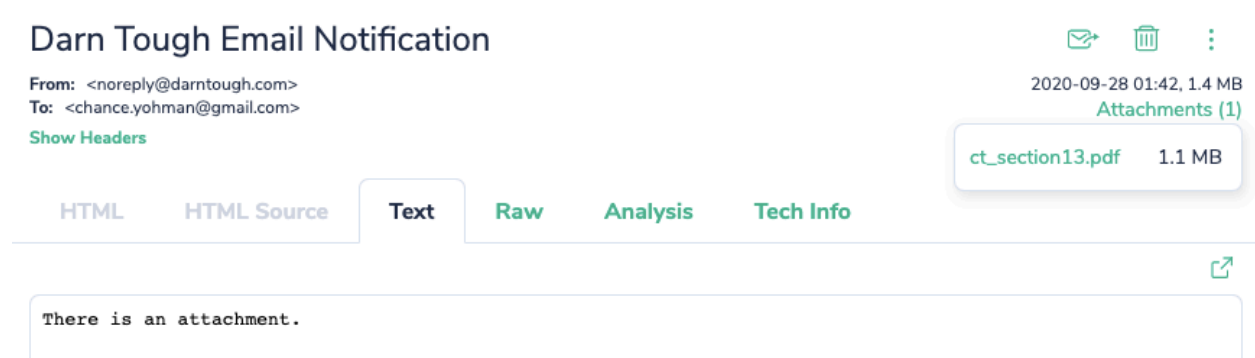
After clicking attachment, clicking on ct_section13.pdf, and opening it in PDF reader of choice(note, I have opened a lot of these– so the filename has the count–1 in it):
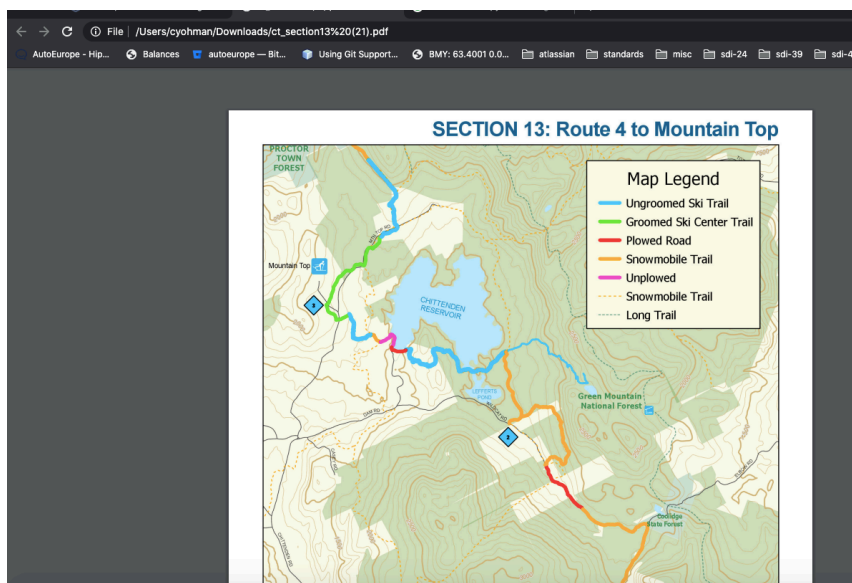
Feel free to inspect and modify any of the tests under "tests", correspondingly inspect modify the data in "tests/data", and re-run the tests.

7) To run the web service from the console your just ran the tests in:

    python api.py

```
(venv) (base) cyohman@Chances-MacBook-Air darnTough % python api.py
 * Serving Flask app "api" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 280-287-497
```
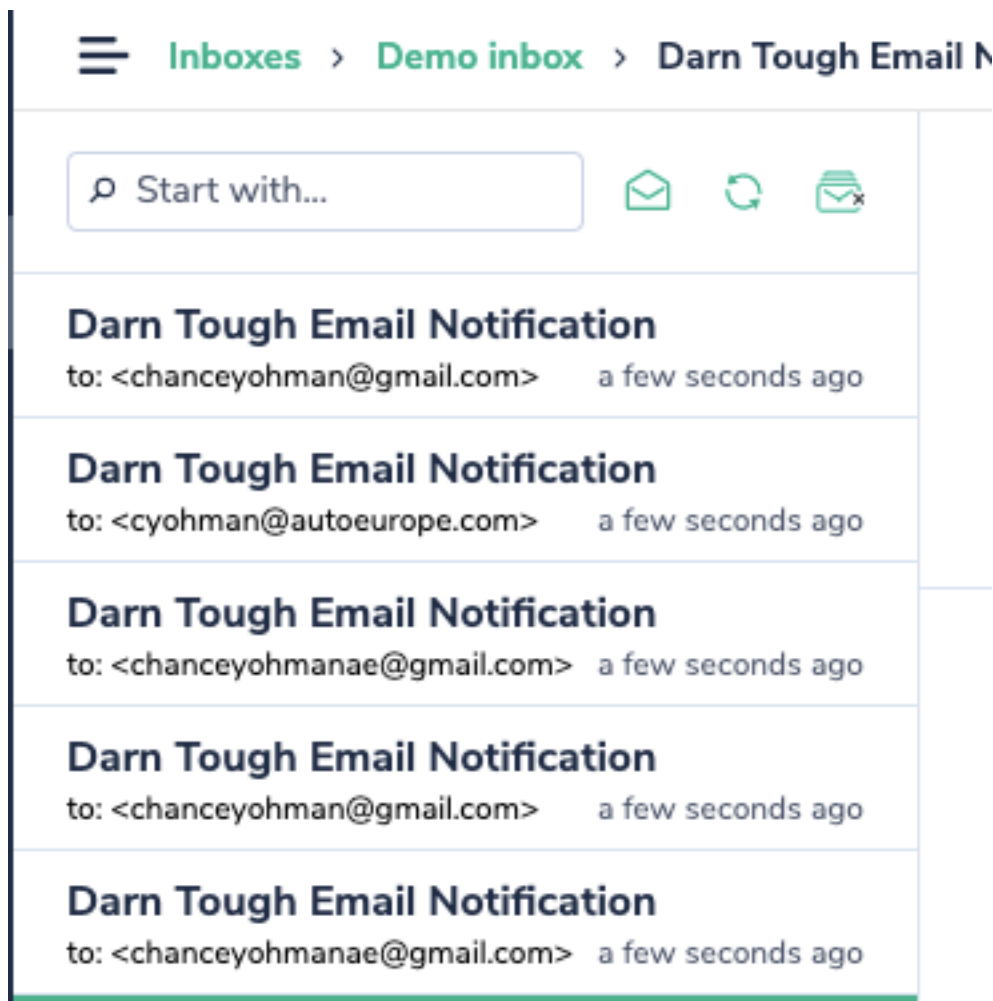
8) In a separate shell in the same directory, run:

    python3 -m venv venv
    source venv/bin/activate (Replacing with Windows command if needed)
    python dtEmail.py

```
(base) cyohman@Chances-MacBook-Air darnTough % python3 -m venv venv

(base) cyohman@Chances-MacBook-Air darnTough % source venv/bin/activate
(venv) (base) cyohman@Chances-MacBook-Air darnTough % python dtEmail.py
Sending email.
Sending email.
Sending email.
Sending email.
Sending email.
(venv) (base) cyohman@Chances-MacBook-Air darnTough %
```

    If you look in mailtrap.io, then you will see the five
    notifications have been sent.

Inboxes > Demo inbox > Darn Tough Email N

Darn Tough Email Notification
to: <chanceyohman@gmail.com>   a few seconds ago

Darn Tough Email Notification
to: <cyohman@autoeurope.com>   a few seconds ago

Darn Tough Email Notification
to: <chanceyohmanae@gmail.com> a few seconds ago

Darn Tough Email Notification
to: <chanceyohman@gmail.com>   a few seconds ago

Darn Tough Email Notification
to: <chanceyohmanae@gmail.com> a few seconds ago

I create a queue based on line entries in the queue file.

    The format is:
    email[required],[optional] file[optiona]

This is not realistic, but for speed of implementation I made this
decision. In reality, there might be a QueueProcessor application
pulling from the Queue Service asynchronously (GET /dequeue) and
posting asynchronously to the email service.

I used asyncio, because it's the asychronous pattern I could adapt the
fastest. I create a series of tasks from the queue to post to
endpoint. I assume all entries in the queue have a valid email address
and if they have a document then the document is valid too (the path
is correct and the file exists in tests/data folder). In reality there
would be error checking of the email address and documents. Also, I

would look at my responses from the service to determine success or failure.

Finally, I wait on the tasks as they are processed.

Here is the queue:

```
cyohman@autoeurope.com, mad_river_glen.jpg
chanceyohman@gmail.com
chanceyohmanae@gmail.com, DT_logo.png
chanceyohmanae@gmail.com
chanceyohman@gmail.com, ct_section13.pdf
~
```

Here is what each of the corresponding emails look like:

Darn Tough Email Notification

From: <noreply@darntough.com>
To: <cyohman@autoeurope.com>
Show Headers

2020-09-28 02:06, 98 KB
Attachments (1)

| HTML | HTML Source | Text | Raw | Analysis | Tech Info |

There is an attachment.

Darn Tough Email Notification

From: <noreply@darntough.com>
To: <chanceyohman@gmail.com>

Show Headers

HTML	HTML Source	**Text**	Raw	Analysis	Tech Info

There is no attachment.

Darn Tough Email Notification

Start with...

Darn Tough Email Notification
to: <chanceyohman@gmail.com> 14 minutes ago

Darn Tough Email Notification
to: <cyohman@autoeurope.com> 14 minutes ago

Darn Tough Email Notification
to: <chanceyohmanae@gmail.com> 14 minutes ago

Darn Tough Email Notification
to: <chanceyohman@gmail.com> 14 minutes ago

Darn Tough Email Notification

From: <noreply@darntough.com>
To: <chanceyohmanae@gmail.com>

Show Headers

Attachments (1)

DT_logo.png	31 KB

HTML	HTML Source	Text	Raw	Analysis	Tech Info

There is an attachment.

Darn Tough Email Notification

From: <noreply@darntough.com>
To: <chanceyohmanae@gmail.com>

2020-09-28 02:06, 327 Bytes

Show Headers

| HTML | HTML Source | **Text** | Raw | Analysis | Tech Info |

```
There is no attachment.
```

Upgrade   chance.yohman@gmail.com

Start with...

Darn Tough Email Notification
to: <chanceyohman@gmail.com>   16 minutes ago

Darn Tough Email Notification
to: <cyohman@autoeurope.com>   16 minutes ago

Darn Tough Email Notification
to: <chanceyohmanae@gmail.com>   16 minutes ago

Darn Tough Email Notification

Darn Tough Email Notification

From: <noreply@darntough.com>
To: <chanceyohman@gmail.com>

2020-09-28 02:06, 1.4 MB
Attachments (1)

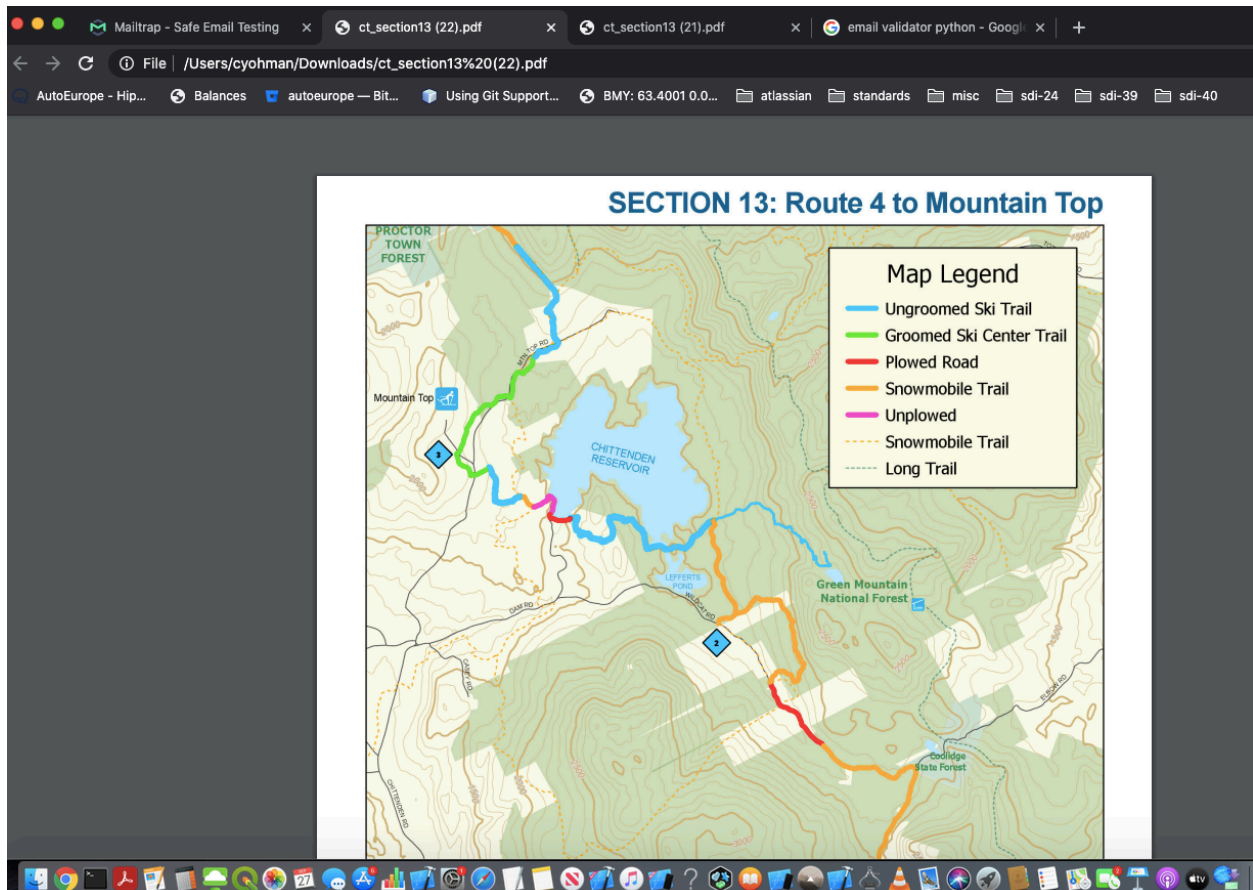ct_section13.pdf   1.1 MB

Show Headers

| HTML | HTML Source | **Text** | Raw | Analysis | Tech Info |

```
There is an attachment.
```

SECTION 13: Route 4 to Mountain Top

Map Legend
- Ungroomed Ski Trail
- Groomed Ski Center Trail
- Plowed Road
- Snowmobile Trail
- Unplowed
- Snowmobile Trail
- Long Trail

That's a wrap.

There are two issues I'll mention with the code. It does not seem to handle .doc or docx. The filetype.guess function on the latter detected it as a zip file. In the interest of time, I decided to note it instead of fixing it. In the real world, this is something that would need to be fixed.

Also, I did notice mailtrap.io, does start dropping messages when the queue if six or more email messages are sent in quick succession. Something along the line of too many emails too fast. Likely a pause would need to be introduced before sending an email. Alternatively, we would need to tweak the mail server if we controlled it. Lastly, we could use an external service to send the email. This was a proof of concept, so I just noted it as something to fix in the real world.