Case for Dataingeniør av Sandra Moen

En Google Analytics Ecommerce Simulator

Casebeskrivelse

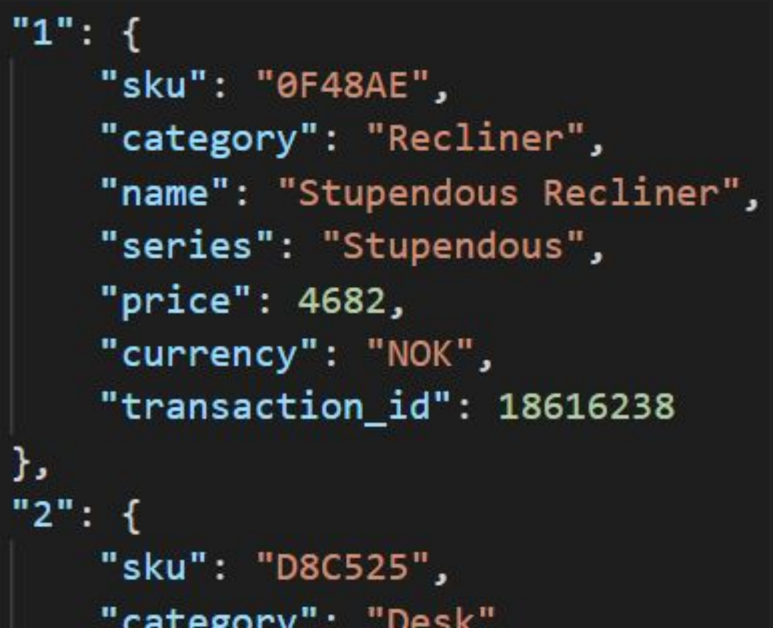
- Fiktiv møbelforhandler
- Ønsker å laste opp salgsdata fra lagerutsalget til Google Analytics

Oppgavebeskrivelse

- Lag et program som sender transaksjoner til Google Analytics
- Bruk datasettet fabfurniture.json som datagrunnlag
    - For hvert element i listen, sende en transaksjon til Google Analytics

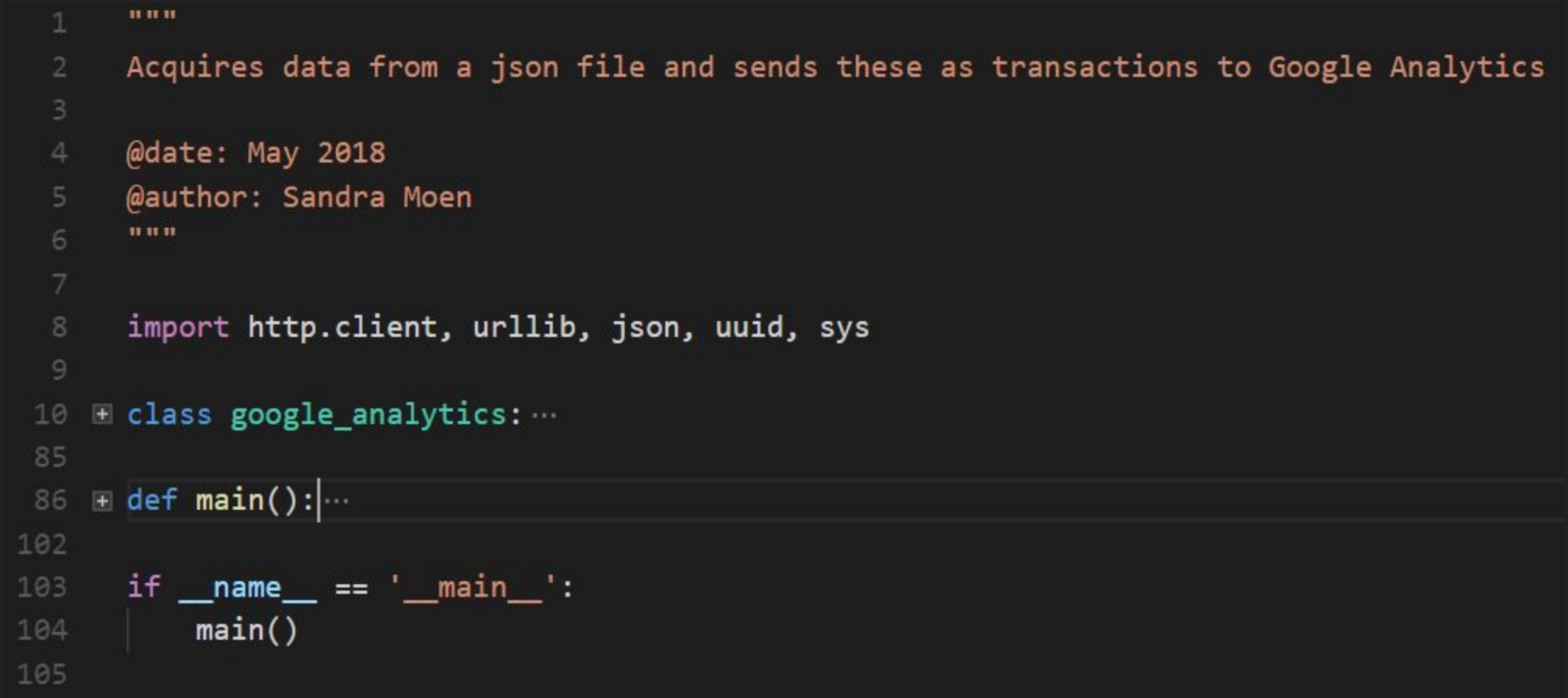
Datasettet

- En liste med transaksjoner i JSON format.
- Hver transaksjon har en unik id, og informasjon om produktet som ble kjøpt
- Hver transaksjon inneholder kun ett produkt

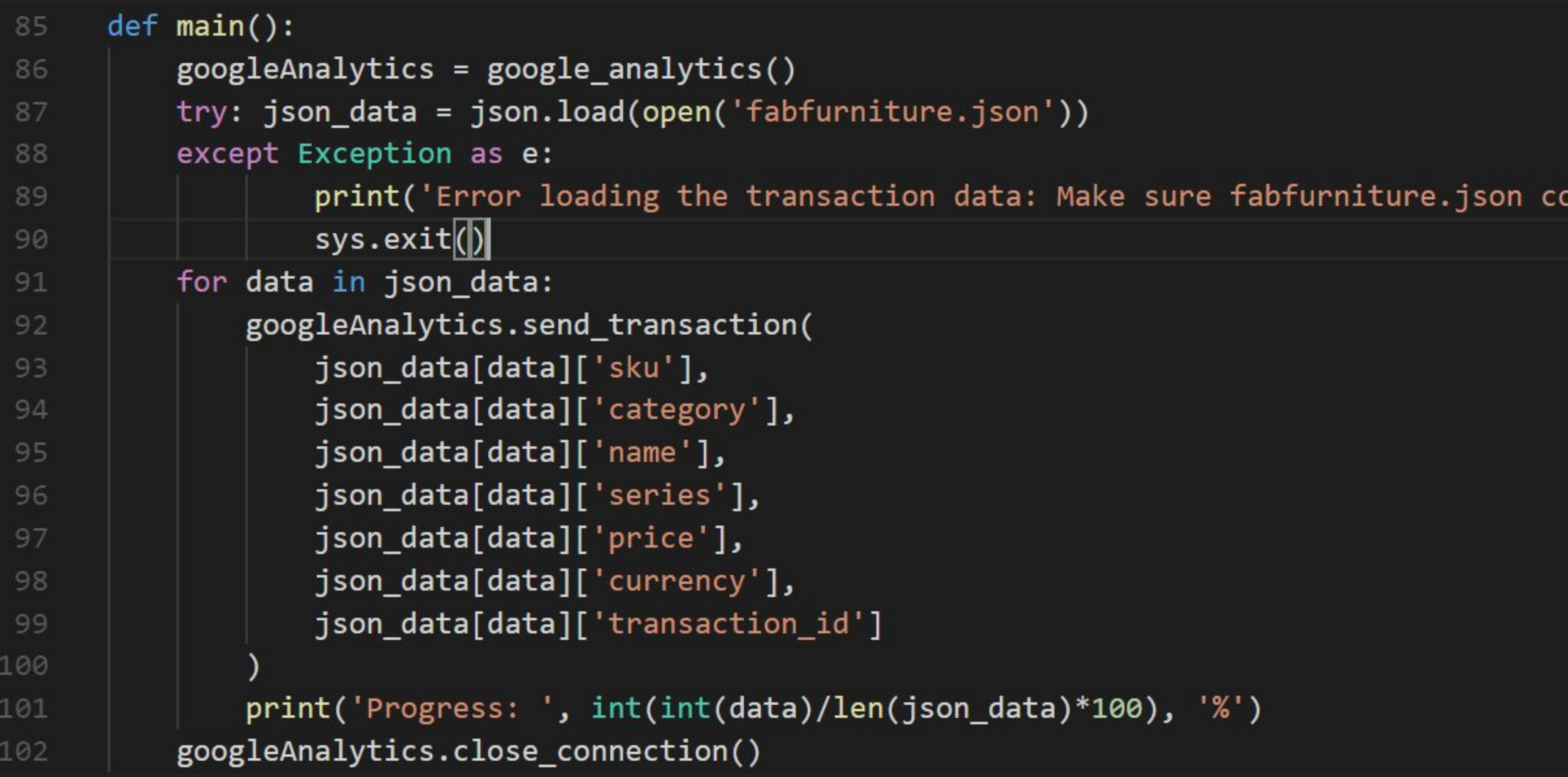


```
"1": {
    "sku": "0F48AE",
    "category": "Recliner",
    "name": "Stupendous Recliner",
    "series": "Stupendous",
    "price": 4682,
    "currency": "NOK",
    "transaction_id": 18616238
},
"2": {
    "sku": "D8C525",
    "category": "Desk"
```

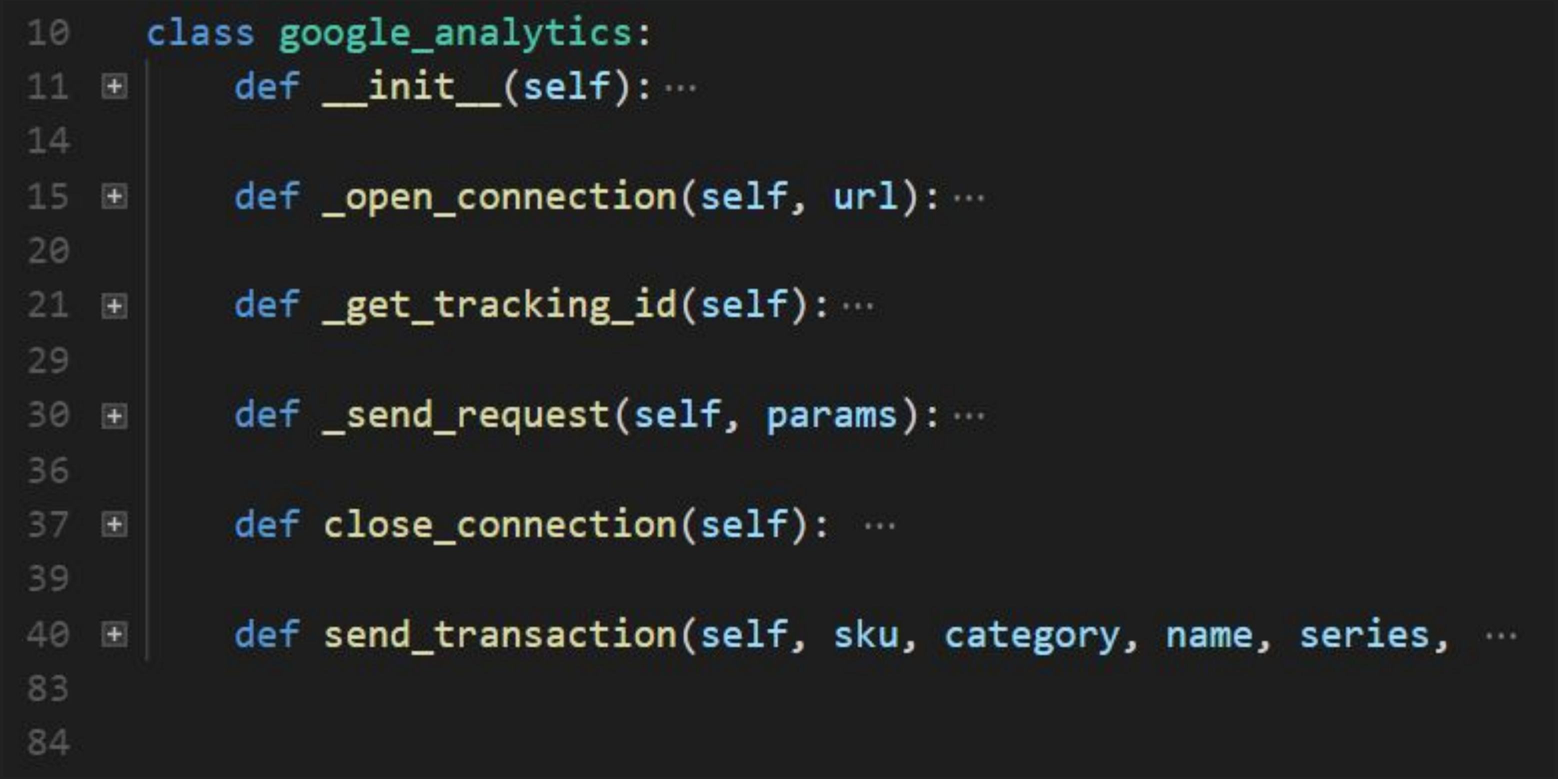
main.py

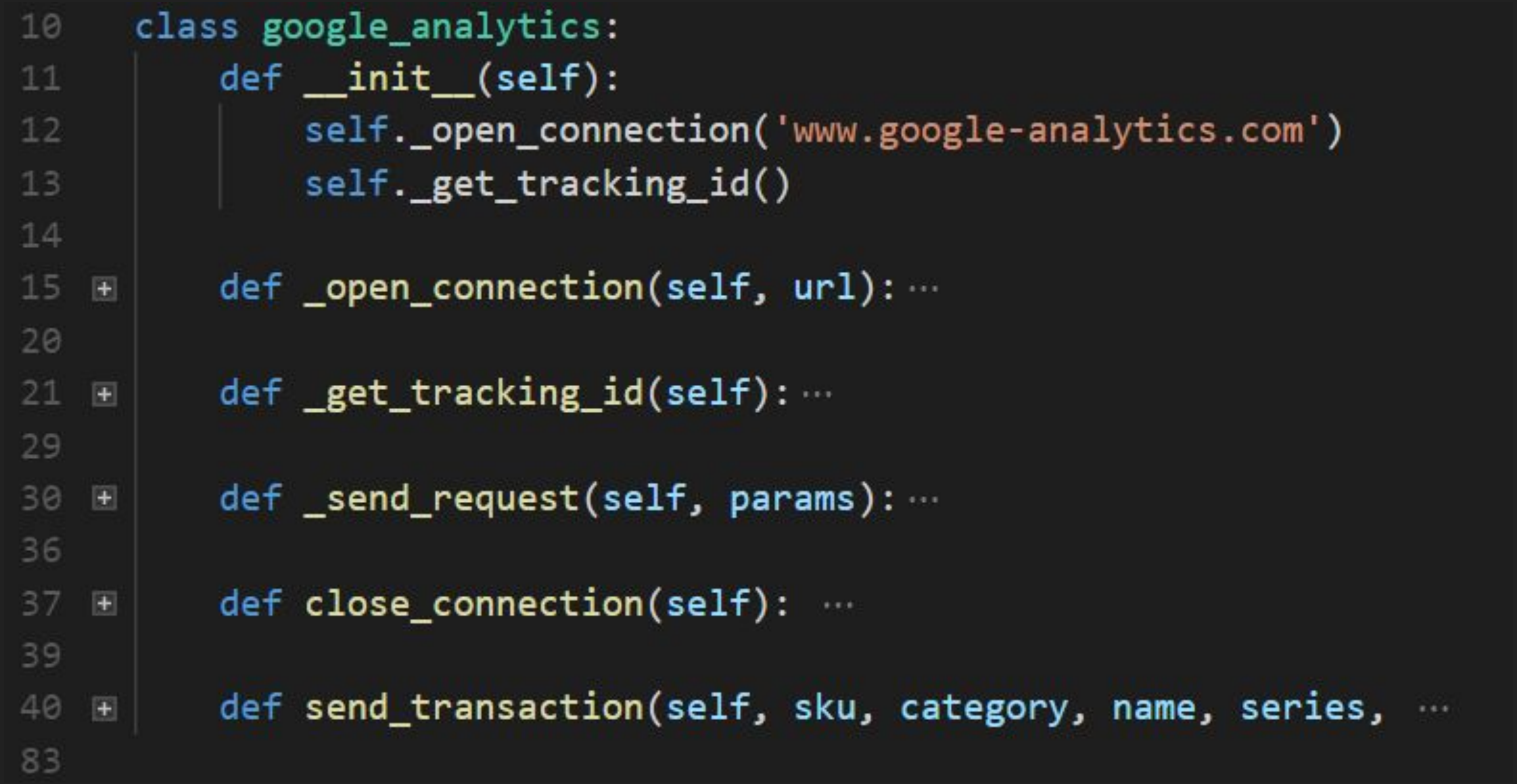


```python
"""
Acquires data from a json file and sends these as transactions to Google Analytics

@date: May 2018
@author: Sandra Moen
"""

import http.client, urllib, json, uuid, sys

class google_analytics:⋯

def main():⋯

if __name__ == '__main__':
    main()

```

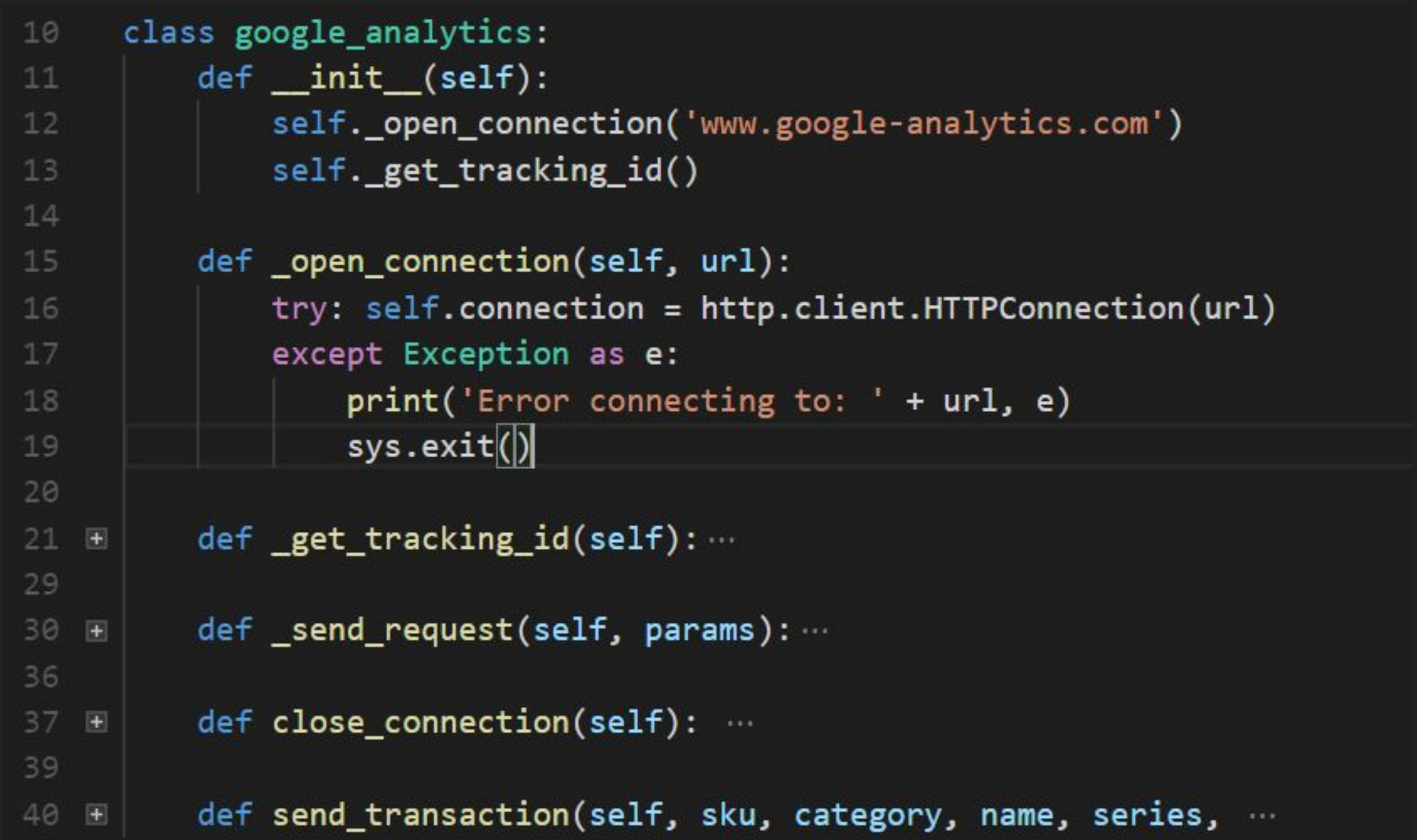


```python
def main():
    googleAnalytics = google_analytics()
    try: json_data = json.load(open('fabfurniture.json'))
    except Exception as e:
            print('Error loading the transaction data: Make sure fabfurniture.json co
            sys.exit()
    for data in json_data:
        googleAnalytics.send_transaction(
            json_data[data]['sku'],
            json_data[data]['category'],
            json_data[data]['name'],
            json_data[data]['series'],
            json_data[data]['price'],
            json_data[data]['currency'],
            json_data[data]['transaction_id']
        )
        print('Progress: ', int(int(data)/len(json_data)*100), '%')
    googleAnalytics.close_connection()
```

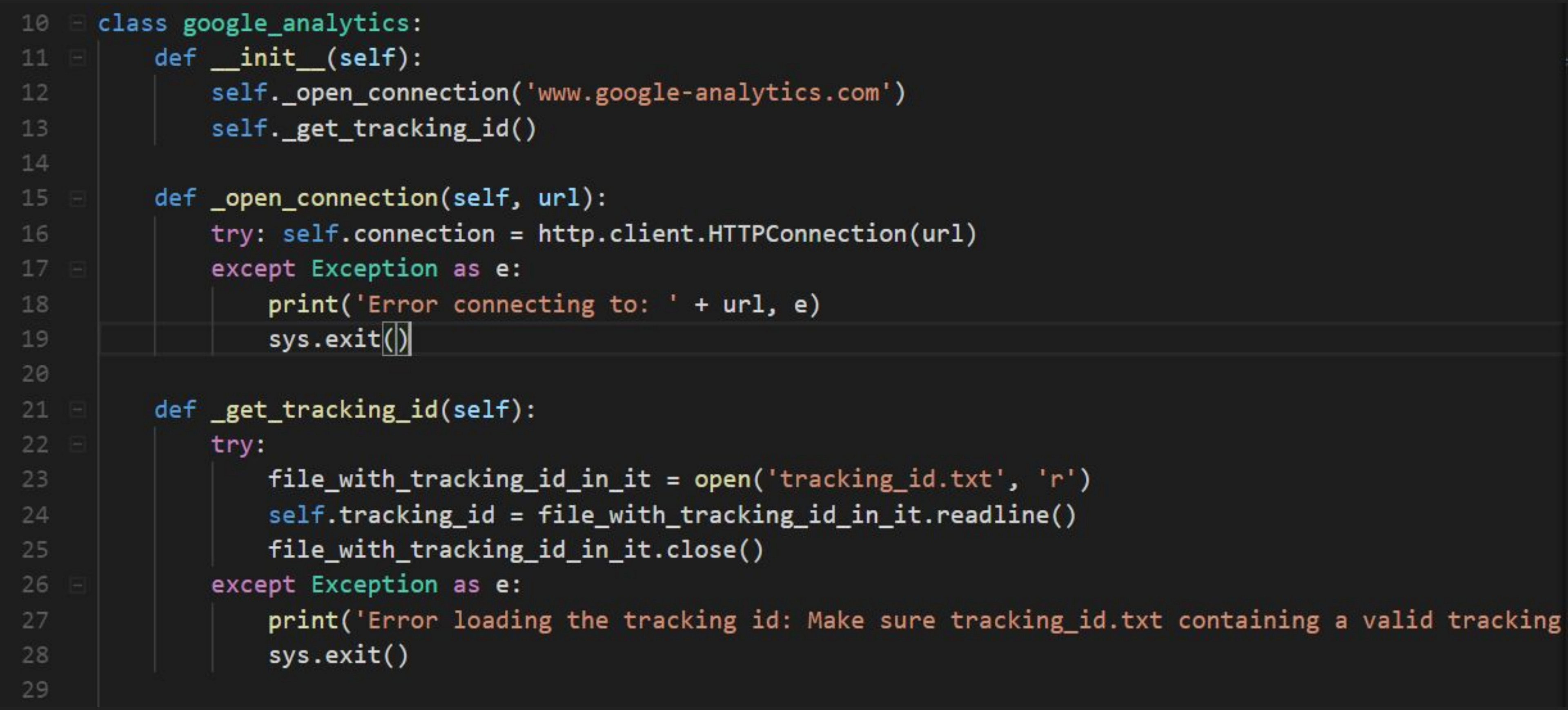

```python
class google_analytics:
    def __init__(self): ⋯

    def _open_connection(self, url): ⋯

    def _get_tracking_id(self): ⋯

    def _send_request(self, params): ⋯

    def close_connection(self):  ⋯

    def send_transaction(self, sku, category, name, series,  ⋯


```



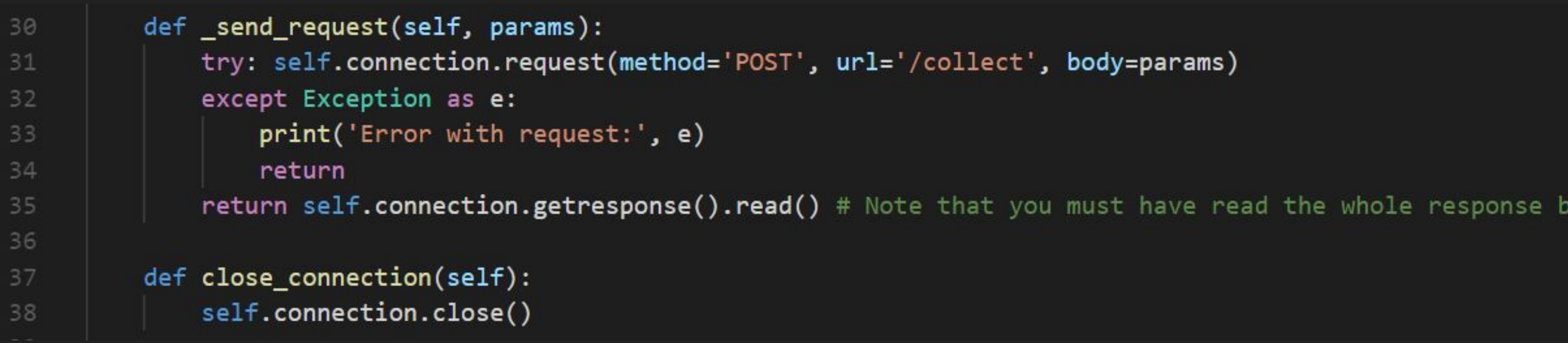
```python
class google_analytics:
    def __init__(self):
        self._open_connection('www.google-analytics.com')
        self._get_tracking_id()

    def _open_connection(self, url):···

    def _get_tracking_id(self):···

    def _send_request(self, params):···

    def close_connection(self): ···

    def send_transaction(self, sku, category, name, series, ···

```



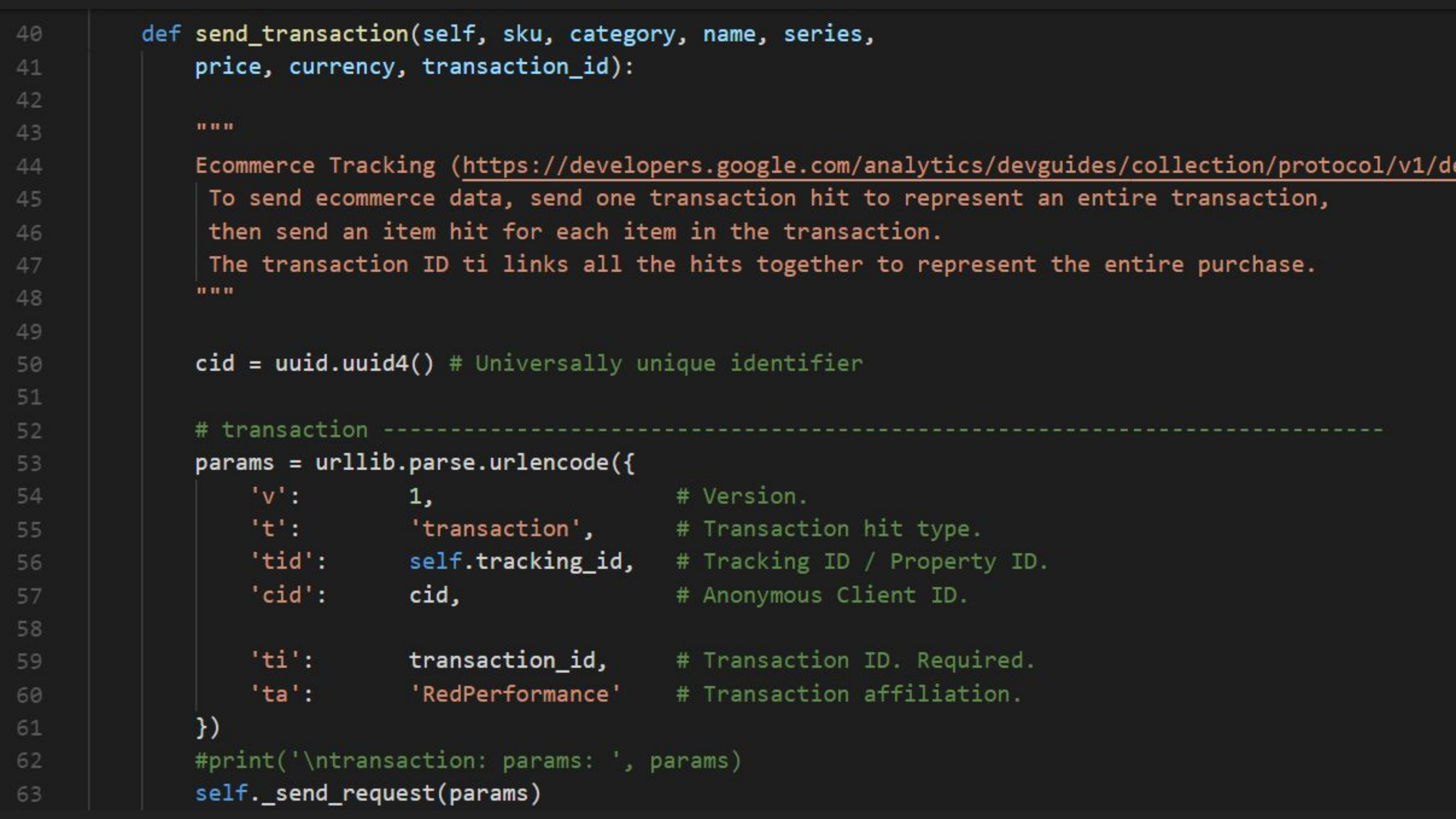
```python
class google_analytics:
    def __init__(self):
        self._open_connection('www.google-analytics.com')
        self._get_tracking_id()

    def _open_connection(self, url):
        try: self.connection = http.client.HTTPConnection(url)
        except Exception as e:
            print('Error connecting to: ' + url, e)
            sys.exit()

    def _get_tracking_id(self):···

    def _send_request(self, params):···

    def close_connection(self): ···

    def send_transaction(self, sku, category, name, series, ···
```



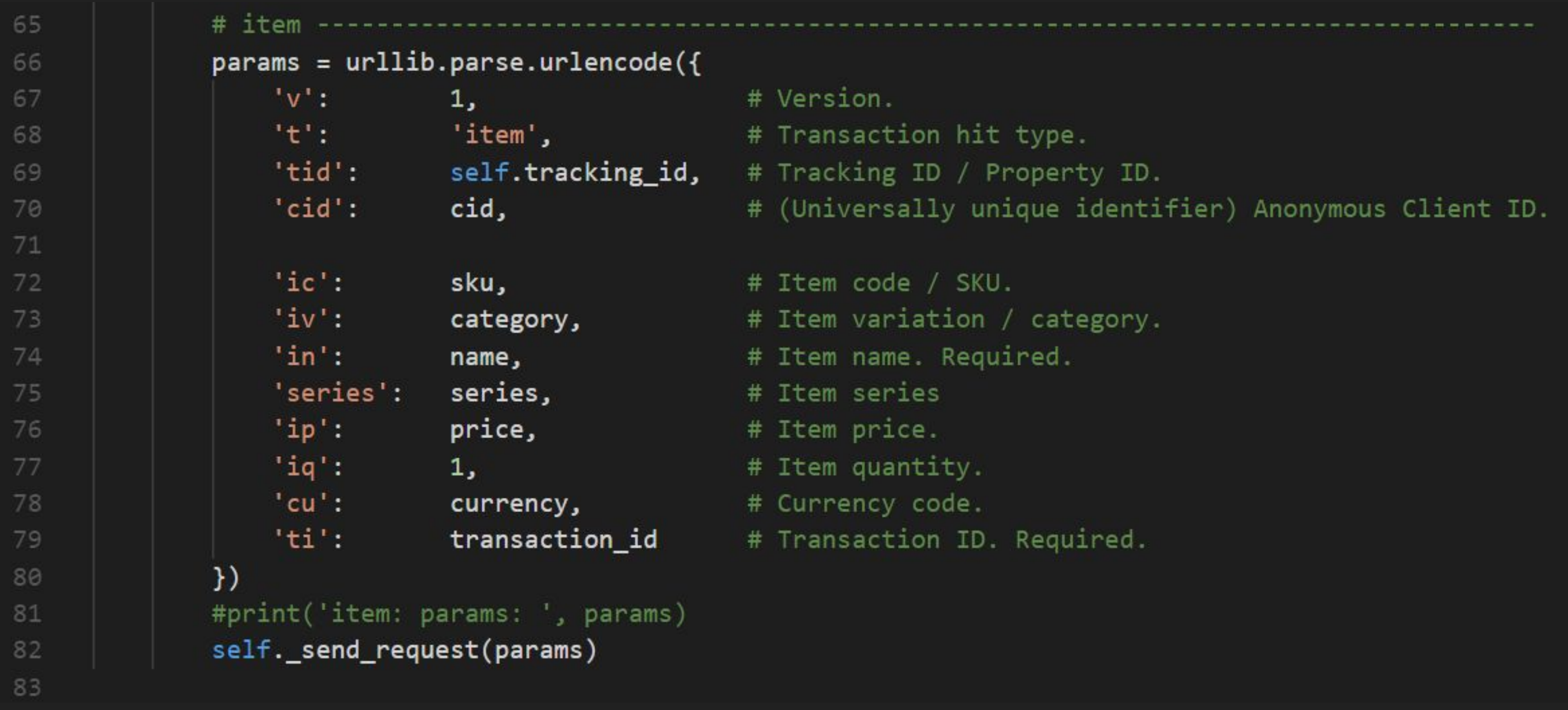
```
class google_analytics:
    def __init__(self):
        self._open_connection('www.google-analytics.com')
        self._get_tracking_id()

    def _open_connection(self, url):
        try: self.connection = http.client.HTTPConnection(url)
        except Exception as e:
            print('Error connecting to: ' + url, e)
            sys.exit()

    def _get_tracking_id(self):
        try:
            file_with_tracking_id_in_it = open('tracking_id.txt', 'r')
            self.tracking_id = file_with_tracking_id_in_it.readline()
            file_with_tracking_id_in_it.close()
        except Exception as e:
            print('Error loading the tracking id: Make sure tracking_id.txt containing a valid tracking
            sys.exit()

```



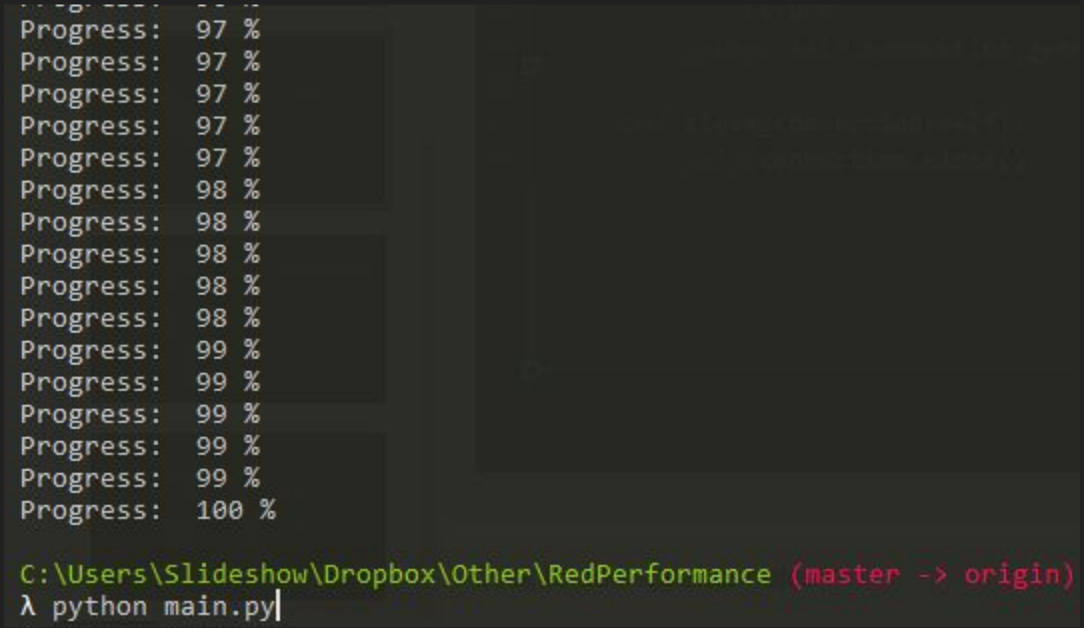
```
    def _send_request(self, params):
        try: self.connection.request(method='POST', url='/collect', body=params)
        except Exception as e:
            print('Error with request:', e)
            return
        return self.connection.getresponse().read() # Note that you must have read the whole response b

    def close_connection(self):
        self.connection.close()
```



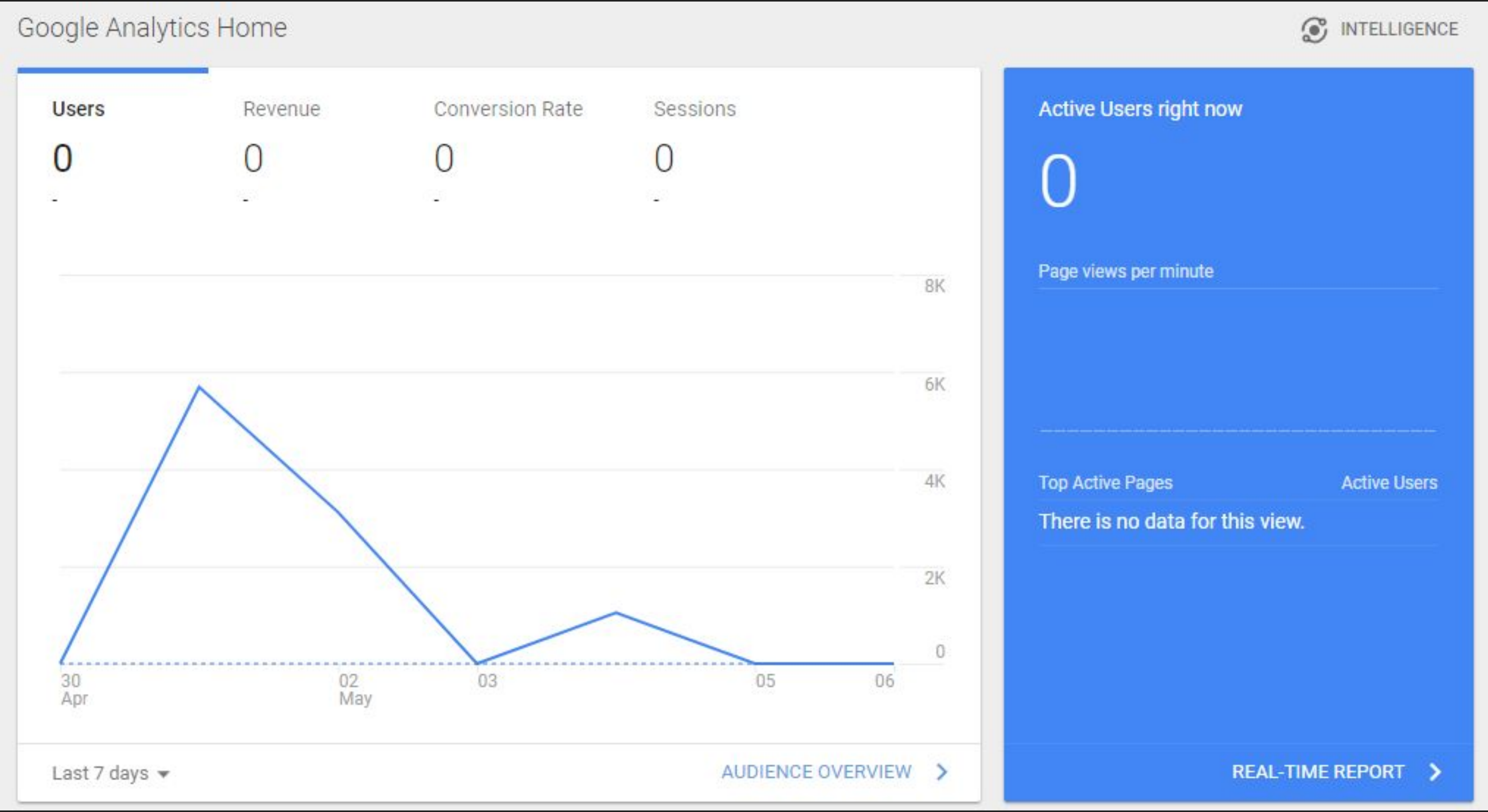
```
    def send_transaction(self, sku, category, name, series,
        price, currency, transaction_id):

        """
        Ecommerce Tracking (https://developers.google.com/analytics/devguides/collection/protocol/v1/de
         To send ecommerce data, send one transaction hit to represent an entire transaction,
         then send an item hit for each item in the transaction.
         The transaction ID ti links all the hits together to represent the entire purchase.
        """


        cid = uuid.uuid4() # Universally unique identifier

        # transaction ---------------------------------------------------------------------------
        params = urllib.parse.urlencode({
            'v':        1,                  # Version.
            't':        'transaction',      # Transaction hit type.
            'tid':      self.tracking_id,   # Tracking ID / Property ID.
            'cid':      cid,                # Anonymous Client ID.

            'ti':       transaction_id,     # Transaction ID. Required.
            'ta':       'RedPerformance'    # Transaction affiliation.
        })
        #print('\ntransaction: params: ', params)
        self._send_request(params)
```



```python
        # item ----------------------------------------------------------------------------------
        params = urllib.parse.urlencode({
            'v':        1,                  # Version.
            't':        'item',             # Transaction hit type.
            'tid':      self.tracking_id,   # Tracking ID / Property ID.
            'cid':      cid,                # (Universally unique identifier) Anonymous Client ID.

            'ic':       sku,                # Item code / SKU.
            'iv':       category,           # Item variation / category.
            'in':       name,               # Item name. Required.
            'series':   series,             # Item series
            'ip':       price,              # Item price.
            'iq':       1,                  # Item quantity.
            'cu':       currency,           # Currency code.
            'ti':       transaction_id      # Transaction ID. Required.
        })
        #print('item: params: ', params)
        self._send_request(params)
```



```
Progress:  97 %
Progress:  97 %
Progress:  97 %
Progress:  97 %
Progress:  97 %
Progress:  98 %
Progress:  98 %
Progress:  98 %
Progress:  98 %
Progress:  98 %
Progress:  99 %
Progress:  99 %
Progress:  99 %
Progress:  99 %
Progress:  99 %
Progress:  100 %

C:\Users\Slideshow\Dropbox\Other\RedPerformance (master -> origin)
λ python main.py
```



Google Analytics Home

INTELLIGENCE

Users	Revenue	Conversion Rate	Sessions
0	0	0	0
-	-	-	-

Last 7 days

AUDIENCE OVERVIEW

Active Users right now

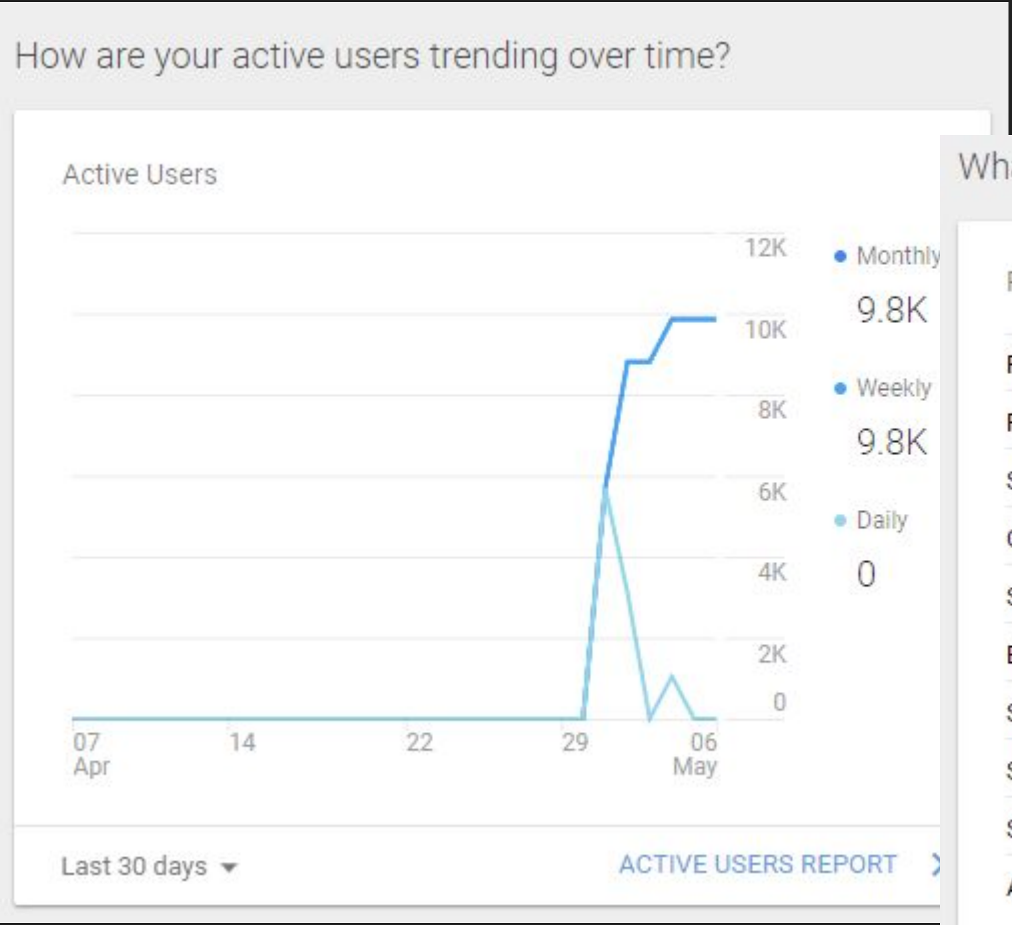
0

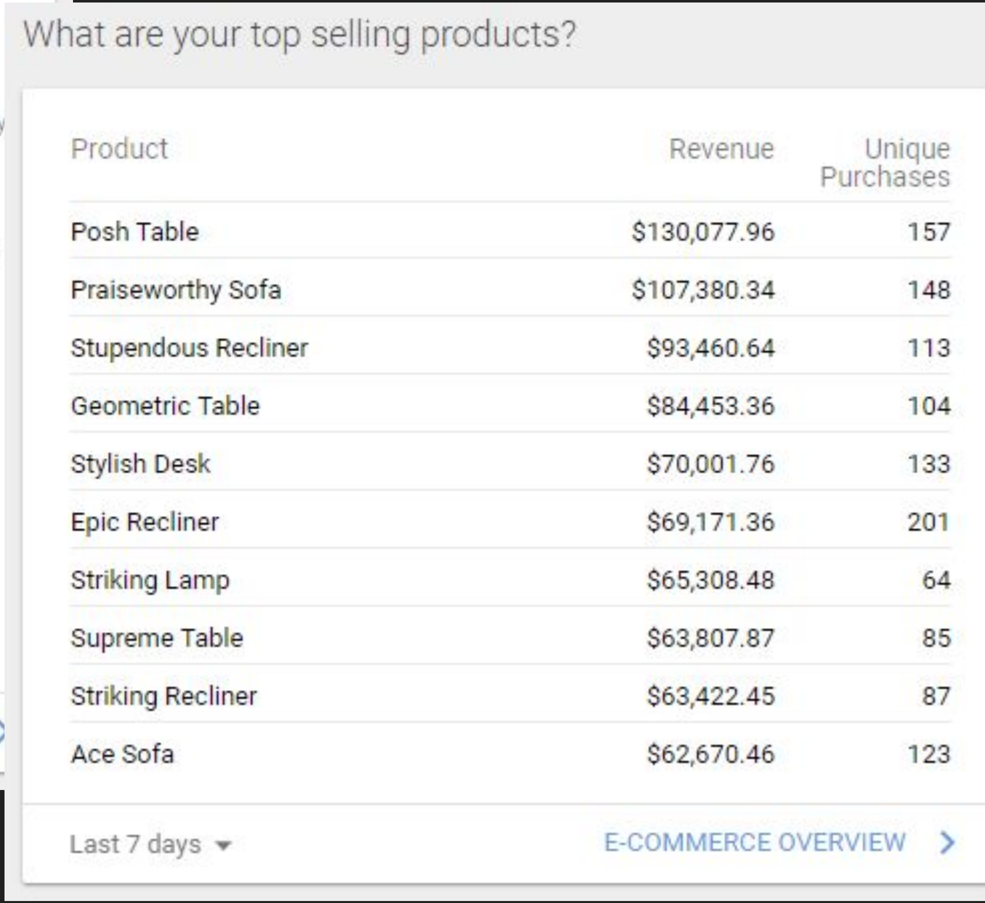
Page views per minute

Top Active Pages	Active Users

There is no data for this view.

REAL-TIME REPORT





What are your top selling products?

Product	Revenue	Unique Purchases
Posh Table	$130,077.96	157
Praiseworthy Sofa	$107,380.34	148
Stupendous Recliner	$93,460.64	113
Geometric Table	$84,453.36	104
Stylish Desk	$70,001.76	133
Epic Recliner	$69,171.36	201
Striking Lamp	$65,308.48	64
Supreme Table	$63,807.87	85
Striking Recliner	$63,422.45	87
Ace Sofa	$62,670.46	123

Last 7 days ▾ E-COMMERCE OVERVIEW ›

How am I so good at this? I've never written code before!
python

light roast comics