

SENDGRID INTEGRATION WITH PYTHON

Date	12 Nov 2022
Team ID	PNT2022TMID39356
Project Name	NUTRITION ASSISTANT APPLICATION

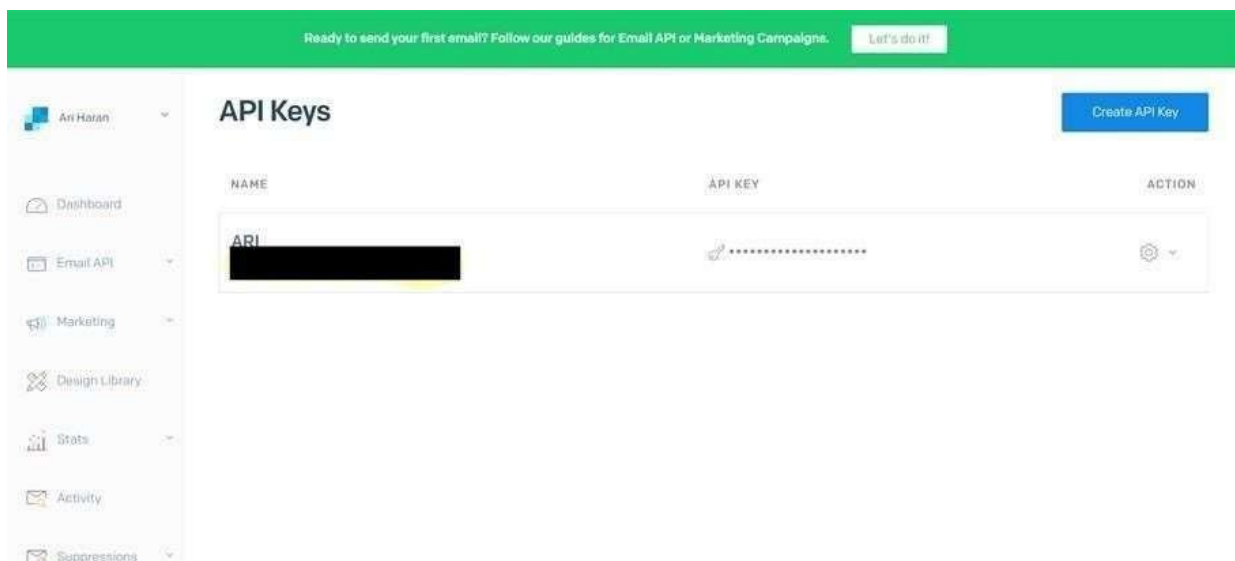
STEP 1:

Requirements:

Python 2.6, 2.7, 3.4 or 3.5.

STEP 2:

Creating an API key



STEP 3:

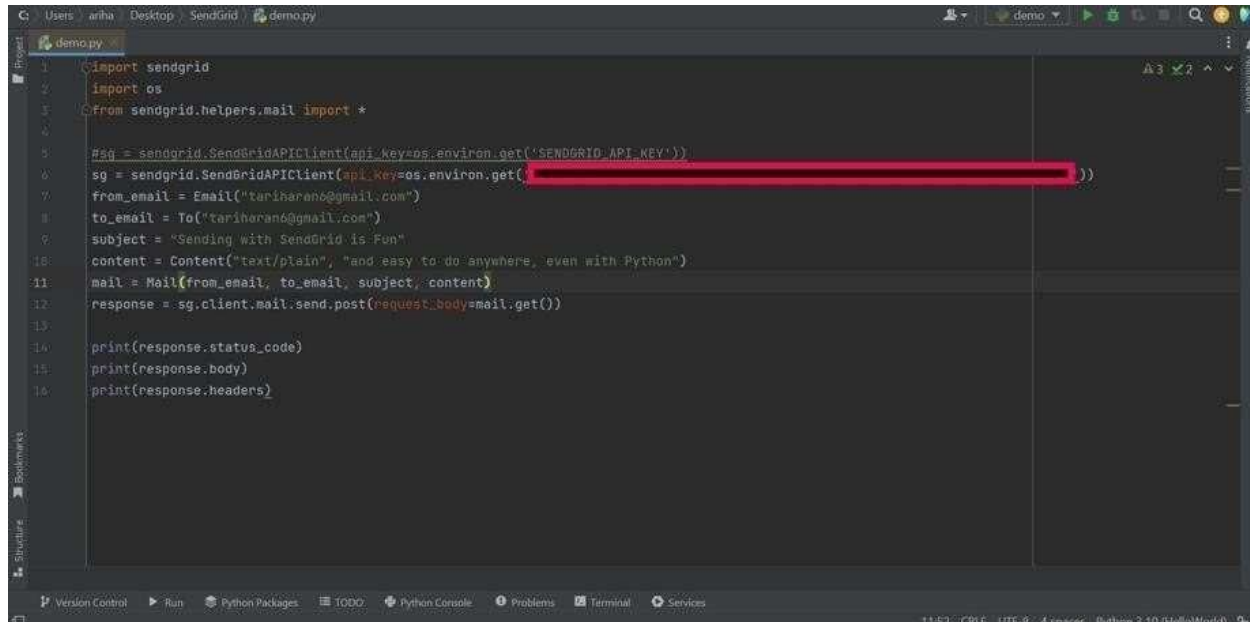
INSTALL

PACKAGE:

> pip install sendgrid

Step 4

SEND EMAIL



The screenshot shows a code editor with a Python script named `demo.py`. The script imports the `sendgrid` and `os` modules, and uses `sendgrid.helpers.mail` to create an email object. It then uses the `sendgrid.SendGridAPIClient` to send the email. The email content is "and easy to do anywhere, even with Python". The script prints the response status code, body, and headers.

```
1 import sendgrid
2 import os
3 from sendgrid.helpers.mail import *
4
5 #sg = sendgrid.SendGridAPIClient(api_key=os.environ.get('SENDGRID_API_KEY'))
6 sg = sendgrid.SendGridAPIClient(api_key=os.environ.get('SENDGRID_API_KEY'))
7 from_email = Email("tariharan@gmail.com")
8 to_email = To("tariharan@gmail.com")
9 subject = "Sending with SendGrid is Fun"
10 content = Content("text/plain", "and easy to do anywhere, even with Python")
11 mail = Mail(from_email, to_email, subject, content)
12 response = sg.client.mail.send.post(request_body=mail.get())
13
14 print(response.status_code)
15 print(response.body)
16 print(response.headers)
```

SENDGRID PYTHON CODE :

```
1 """HTTP Client library"""
2 import json
3 import logging
4 from .exceptions import handle_error
5
6 try:
7     # Python 3
8     import urllib.request as urllib
9     from urllib.parse import urlencode
10     from urllib.error import HTTPError
11 except ImportError:
12     # Python 2
```

```
1 import os
2 from sendgrid import SendGridAPIClient
3 from sendgrid.helpers.mail import Mail
4
5 message = Mail(
6     from_email='from_email@example.com',
7     to_emails='to@example.com',
8     subject='Sending with Twilio SendGrid is Fun',
9     html_content='<strong>and easy to do anywhere, even with
10 Python</strong>')
11
12 try:
13     sg = SendGridAPIClient(os.environ.get('SENDGRID_API_KEY'))
14     response = sg.send(message)
15     print(response.status_code)
16     print(response.body) 15 print(response.headers) 16 except Exception as
17 e:
18     print(e.message)
```

```
import urllib2 as urllib
```

```
14     from urllib2 import HTTPError
15     from urllib import urlencode
16
17     _logger = logging.getLogger( name )
18
19
20     class Response(object):
21         """Holds the response from an API call.""" 22
22
23         def init (self, response):
24             """
25             :param response: The return value from a
26             open call
27             on a urllib.build_opener()
28             :type response: urllib response object
29             """
30             self._status_code = response.getcode()
31             self._body = response.read()
32             self._headers = response.info()
33
34         @property
```

```
34     def status_code(self):
35         """
36         :return: integer, status code of API call
37         """
38         return self._status_code
39
40     @property
41     def body(self):
42         """
43         :return: response from the API
44         """
45         return self._body
46
47     @property
```

```
48     def headers(self):
49         """
50         :return: dict of response headers
51         """
52         return self._headers
53
54     @property
55     def to_dict(self):
56         """
57         :return: dict of response from the API
58         """
59         if self.body:
60             return json.loads(self.body.decode('utf-8'))
61         else:
62             return None
63
64
65 class Client(object):
66     """Quickly and easily access any REST or REST-like API.""" 67
68     # These are the supported HTTP verbs
```

```

69     methods = {'delete', 'get', 'patch', 'post', 'put'} 70
71     def init (self,
72         host,
73         request_headers=None,
74         version=None,
75         url_path=None,
76         append_slash=False, 77         timeout=None):
78         """
79         :param host: Base URL for the api. (e.g.
80         https://api.sendgrid.com)
81         :type host: string
82         :param request_headers: A dictionary of the headers you want

```

```

83         :type request_headers: dictionary
84         :param version: The version number of the
85         API.
86         Subclass _build_versioned_url for custom
87         behavior.
88         Or just pass the version as part of the URL
89         (e.g. client._("/v3"))
90         :type version: integer
91         :param url_path: A list of the url path
92         segments
93         :type url_path: list of strings
94         """
95         self.host = host
96         self.request_headers = request_headers or {}
97         self._version = version
98         # _url_path keeps track of the dynamically
99         built url
100         self._url_path = url_path or []

```

```
97             # APPEND SLASH set
98             self.append_slash = append_slash
99             self.timeout = timeout
100
101     def _build_versioned_url(self, url):
```



```
102         """Subclass this function for your own needs.
103         Or just pass the version as part of the URL
104         (e.g. client._('/v3'))
105         :param url: URI portion of the full URL being requested
106         :type url: string
107         :return: string
108         """
109         return '{}{}/v{}{}'.format(self.host, str(self._version),
110                                     url)
111
112     def _build_url(self, query_params):
113         """Build the final URL to be passed to urllib
114         :param query_params: A dictionary of all the query
```

```
115         :type query_params: dictionary
116         :return: string
117         """
118         url = ''
119         count = 0
120         while count < len(self._url_path):
121             url += '/{}'.format(self._url_path[count])
122             count += 1
123
124         # add slash
125         if self.append_slash:
126             url += '/'
127
128         if query_params:
129             url_values = urlencode(sorted(query_params.items()), True)
130             url = '{}?{}'.format(url, url_values)
131
132         if self._version:
133             url = self._build_versioned_url(url)
134         else:
135             url = '{}{}'.format(self.host, url)
136         return url
137
138     def _update_headers(self, request_headers):
139         """Update the headers for the request
140
141         :param request_headers: headers to set for the API call
```

```
142         :type request_headers: dictionary
143         :return: dictionary
144         """
145         self.request_headers.update(request_headers)
146
147     def _build_client(self, name=None):
```

```

148         """Make a new Client object
149
150         :param name: Name of the url segment
151         :type name: string
152         :return: A Client object
153         """
154         url_path = self._url_path + [name] if name else
            self._url_path
155         return Client(host=self.host,
156                       version=self._version,
157                       request_headers=self.request_headers,
158                       url_path=url_path,
159                       append_slash=self.append_slash,
160                       timeout=self.timeout)
161
162         def make_request(self, opener, request,
163                           timeout=None):
164             """Make the API call and return the response. is
165             This separated into testing.
166             it's own function, so we can mock it easily for
167
168         :param opener:

```

```
167         :type opener:
168         :param request: url payload to request
169
170         :type request: urllib.Request object
171
172         :param timeout: timeout value or None
173
174         :type timeout: float
175
176         :return: urllib response
177
178         """
179
180         timeout = timeout or self.timeout
181
182         try:
183             return opener.open(request, timeout=timeout)
184
185         except HTTPError as err:
186             exc = handle_error(err)
187
188             exc.cause ____ = None
189
190         _logger.debug('{method} Response: {status}
```

```
        :return: string version
        """
214         self._version = args[0]
215         return self._build_client()
216         return get_version
217
218         # We have reached the end of the method chain, make the
API call
219         if name in self.methods:
220             method = name.upper()
221
222             def http_request(
223                 request_body=None,
224                 query_params=None,
225                 request_headers=None,
226                 timeout=None,
227                 **_):
228                 """Make the API call
229
230                 :param timeout: HTTP request timeout. Will be
propagated to
231                 urllib client
                :type timeout: float
```

```
232             :param request_headers: HTTP headers. Will be
merged into
233             current client object state
234             :type request_headers: dict
235             :param query_params: HTTP query parameters
236             :type query_params: dict
237             :param request_body: HTTP request body
238             :type request_body: string or json-serializable
object
239             :param kwargs:
240             :return: Response object
241             """
242             if request_headers:
```

```
243         self._update_headers(request_headers)
244
245         if request_body is None:
246             data = None
247         else:
248             # Don't serialize to a JSON formatted str
249             # if we don't have a JSON Content-Type
250             if 'Content-Type' in self.request_headers and \
251                 self.request_headers['Content-Type'] != \
252                 'application/json':
253                 data = request_body.encode('utf-8')
254             else:
255                 self.request_headers.setdefault(
256                     'Content-Type', 'application/json')
257                 data =
258                 json.dumps(request_body).encode('utf-8')
259
260         opener = urllib.build_opener()
261         request = urllib.Request(
262             self._build_url(query_params),
263             headers=self.request_headers,
264             data=data,
265         )
266         request.get_method = lambda: method
267
268         _logger.debug('[method] Request: {url}'.format(
269             method=method,
270             url=request.get_full_url()))
271         if request.data:
272             _logger.debug('PAYLOAD: {data}'.format(
273                 data=request.data))
274         _logger.debug('HEADERS: {headers}'.format(
275             headers=request.headers))
```



```
response = Response(
    self.make_request(opener, request,
        timeout=timeout)
278         )

279
280         _logger.debug('{method} Response: {status}
    {body}'.format(
281             method=method,
282             status=response.status_code,
283             body=response.body))
284
285         return response
286
287     return http_request 288
    else:
289         # Add a segment to the URL
290         return self._(name)
291
292     def getstate (self):
293         return self. dict
294
295     def setstate (self, state):
```