SENDGRID INTEGRATION WITH PYTHON

Date	15 Nov 2022
Team ID	PNT2022TMID44798
Project Name	SKILL/JOB RECOMMENDED APP

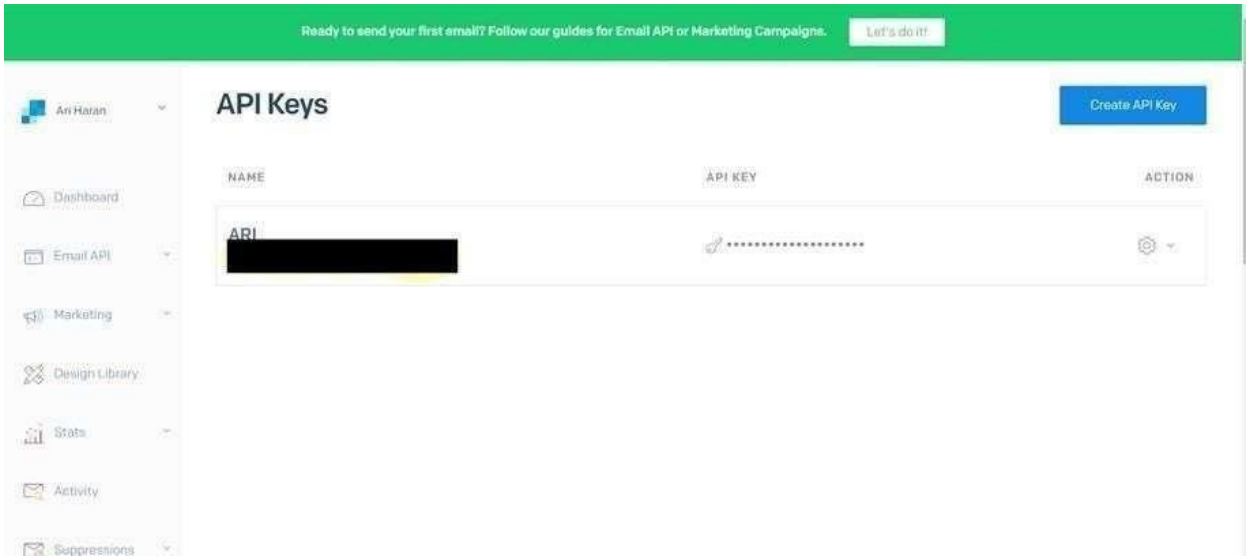
STEP 1:

Requirements:

Python 2.6, 2.7, 3.4 or 3.5.

STEP 2:

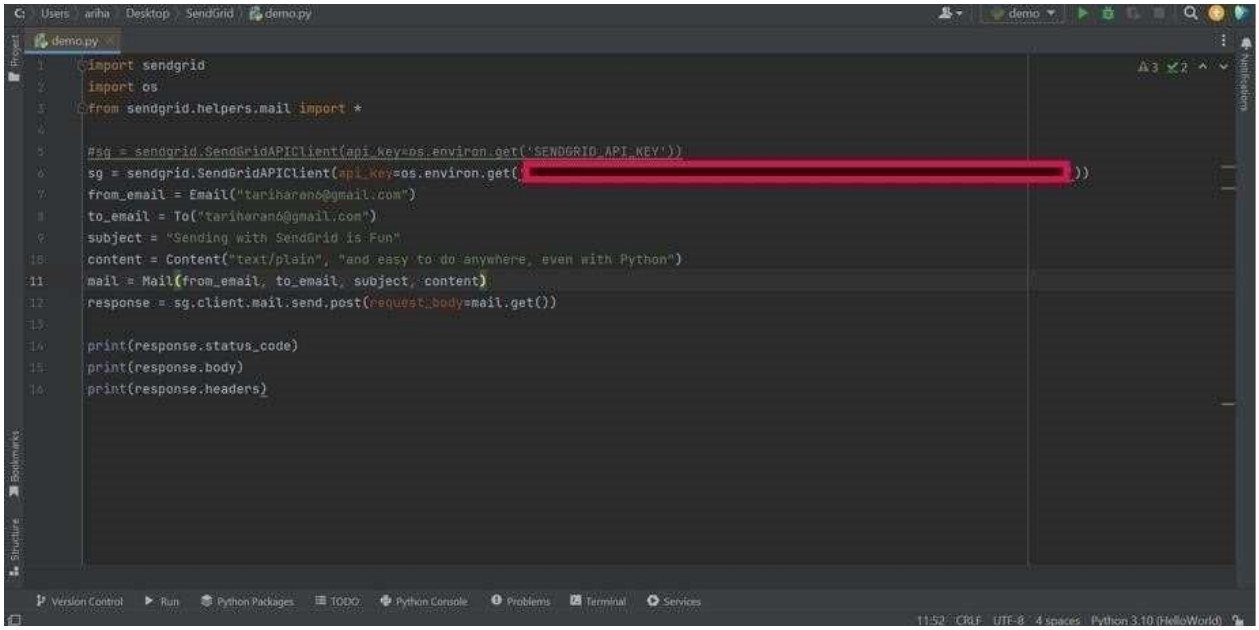
Creating an API key



STEP 3: INSTALL PAKAGE: > pip install sendgrid

SETP 4:

SEND EMAIL



SENDGRID PYTHON CODE :

```
"""HTTP Client library"""
import json
import logging
from .exceptions import handle_error

    try:
    # Python 3
    import urllib.request as urllib
    from urllib.parse import urlencode
    from urllib.error import HTTPError
    except ImportError:
    # Python 2
```

```
import os
from sendgrid import SendGridAPIClient
from sendgrid.helpers.mail import Mail



    message = Mail(
    from_email='from_email@example.com',

    to_emails='to@example.com',

    subject='Sending with Twilio SendGrid is Fun',

    html_content='<strong>and easy to do anywhere, even with
    Python</strong>')
    try:

    sg = SendGridAPIClient(os.environ.get('SENDGRID_API_KEY'))

    response = sg.send(message)
```

```
13      print(response.status_code)


14      print(response.body) 15 print(response.headers) 16 except Exception as
        e:

17      print(e.message)
```

HTTP CLIENT PROGRAM:

```
        import urllib2 as urllib
```

```
    from urllib2 import HTTPError
    from urllib import urlencode

_logger = logging.getLogger( name )


  class Response(object):
  """Holds the response from an API call.""" 22
                        def init (self, response):
                        """
                        :param response: The return value from a
                        open call
                        on a urllib.build_opener()
                        :type response: urllib response object
                        """
                        self._status_code = response.getcode()
                        self._body = response.read()
                        self._headers = response.info()

@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.
    https://api.sendgrid.com)
 80         :type host: string
 81         :param request_headers: A dictionary of the headers you want
82                                  applied on all calls
 83                          :type request_headers: dictionary
 84
                             :param version: The version number of the
                             API.
 85                          Subclass _build_versioned_url for custom
    behavior.
 86                          Or just pass the version as part of the URL
 87                          (e.g. client._("/v3"))
 88                          :type version: integer
 89
                             :param url_path: A list of the url path
                             segments
 90                          :type url_path: list of strings
```

```
91                          """
92                          self.host = host
```

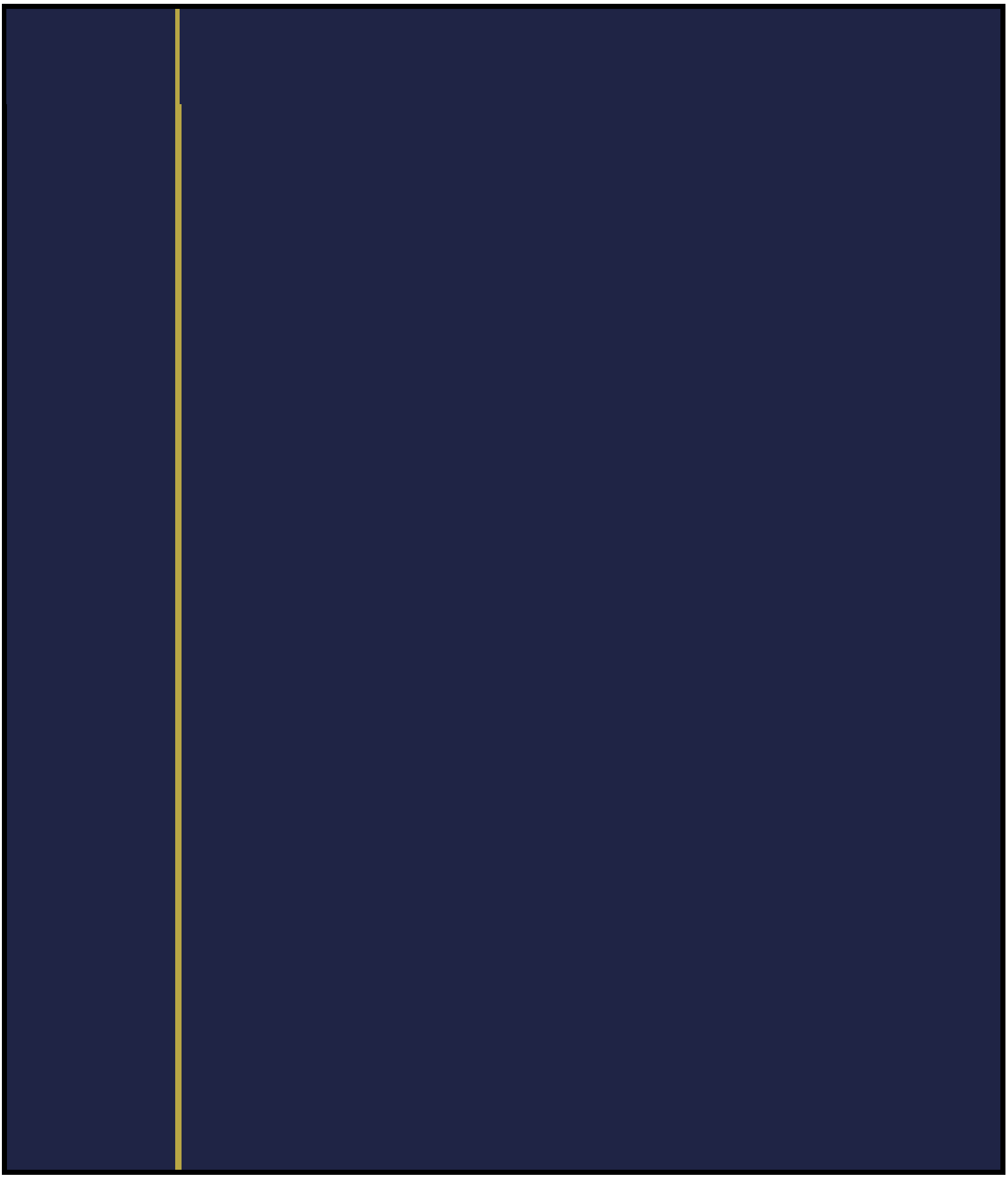
```
93                          self.request_headers = request_headers or {}

94                          self._version = version
95                          # _url_path keeps track of the dynamically
                            built url
96                          self._url_path = url_path or []
```

```
                        # APPEND SLASH set
                        self.append_slash = append_slash
                        self.timeout = timeout
def _build_versioned_url(self, url):
```

```
            """Subclass this function for your own needs.
            Or just pass the version as part of the URL
            (e.g. client._('/v3'))
            :param url: URI portion of the full URL being requested
            :type url: string
            :return: string
            """
```

```
109             return '{}/v{}{}'.format(self.host, str(self._version),
                url)
110
111         def _build_url(self, query_params):
112         """Build the final URL to be passed to urllib
113
114         :param query_params: A dictionary of all the query
```

```
parameters
```



```
        :type query_params: dictionary
        :return: string
        """
        url = ''
        count = 0
        while count < len(self._url_path):
        url += '/{}'.format(self._url_path[count])
        count += 1

        # add slash
        if self.append_slash:
        url += '/'

        if query_params:
        url_values = urlencode(sorted(query_params.items()), True)
        url = '{}?{}'.format(url, url_values)

        if self._version:
        url = self._build_versioned_url(url)
        else:
        url = '{}{}'.format(self.host, url)
        return url

    def _update_headers(self, request_headers):
```

```
        """Update the headers for the request

        :param request_headers: headers to set for the API call
        :type request_headers: dictionary
        :return: dictionary
        """
        self.request_headers.update(request_headers)
```

```
    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,
                timeout=None):
163             """Make the API call and return the response.    is
                This separated into                              testing.
164             it's own function, so we can mock it easily for
165
166          :param opener:
167             :type opener:
168
                :param request: url payload to request
169             :type request: urllib.Request object
170
                :param timeout: timeout value or None
171
                :type timeout: float
```

```
172
                :return: urllib response
173
                """
174
                timeout = timeout or self.timeout
175
                try:
176
                return opener.open(request, timeout=timeout)
177
                except HTTPError as err:


178             exc = handle_error(err)


179             exc. cause = None


180             _logger.debug('{method} Response: {status}
```

```
212
                    :return:  string , version
213                  """
```

```
                self._version = args[0]
                return self._build_client()
                return get_version

            # We have reached the end of the method chain, make the API call  if name in self.methods:
            method = name.upper()

                def http_request(
                request_body=None,
                query_params=None,
                request_headers=None,
                timeout=None,
                **_):
                """Make the API call
                :param timeout: HTTP request timeout. Will be propagated to  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 =
    json.dumps(request_body).encode('utf-8')
258
259                 opener = urllib.build_opener()
260                 request = urllib.Request(
261                     self._build_url(query_params),
262                     headers=self.request_headers,
263                     data=data,
264                 )
265                 request.get_method = lambda: method
266
267                 _logger.debug('{method} Request: {url}'.format(
268                     method=method,
269                     url=request.get_full_url()))
270                 if request.data:
271                     _logger.debug('PAYLOAD: {data}'.format(
272                         data=request.data))
273                 _logger.debug('HEADERS: {headers}'.format(
274                     headers=request.headers))
275
```

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

                    logger.debug('{method} Response: {status}
    {body}'.format(
                    method=method,
                    status=response.status_code,
                    body=response.body))

                return response

    return http_request 288
    else:
            # Add a segment to the URL
            return self._(name)

        def getstate (self):
        return self. dict

    def setstate (self, state):
```