SENDGRID INTEGRATION WITH PYTHON

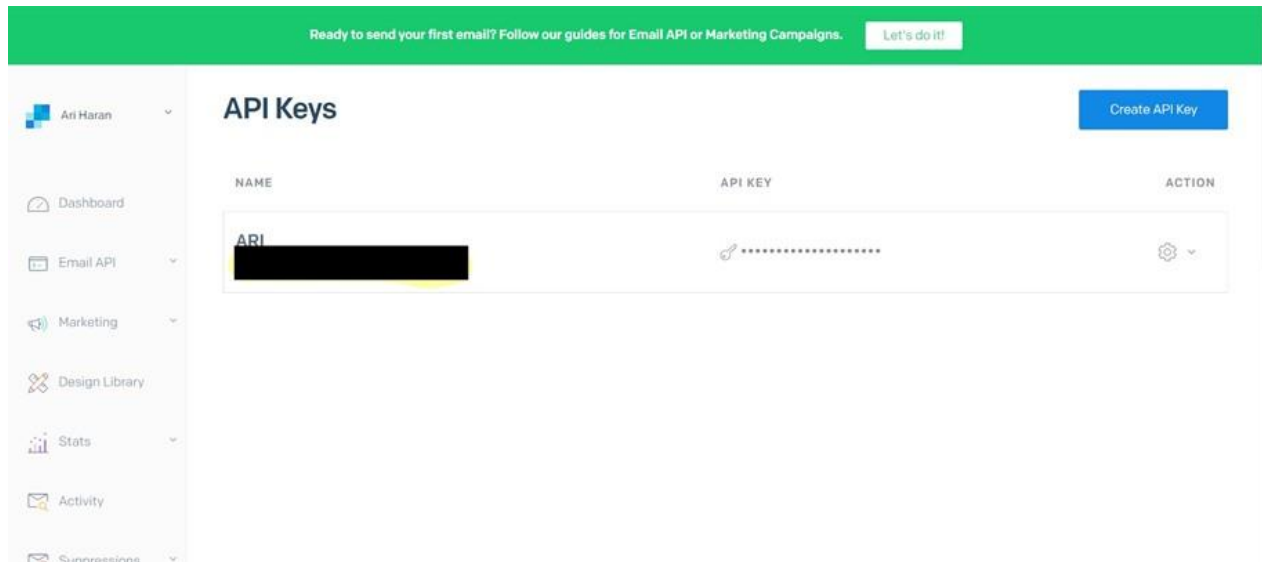| Date	04 Nov 2022
Team ID	PNT2022TMID42524
Project Name	NUTRITION ASSISTANT APPLICATION

STEP 1:

REQUIREMENTS:

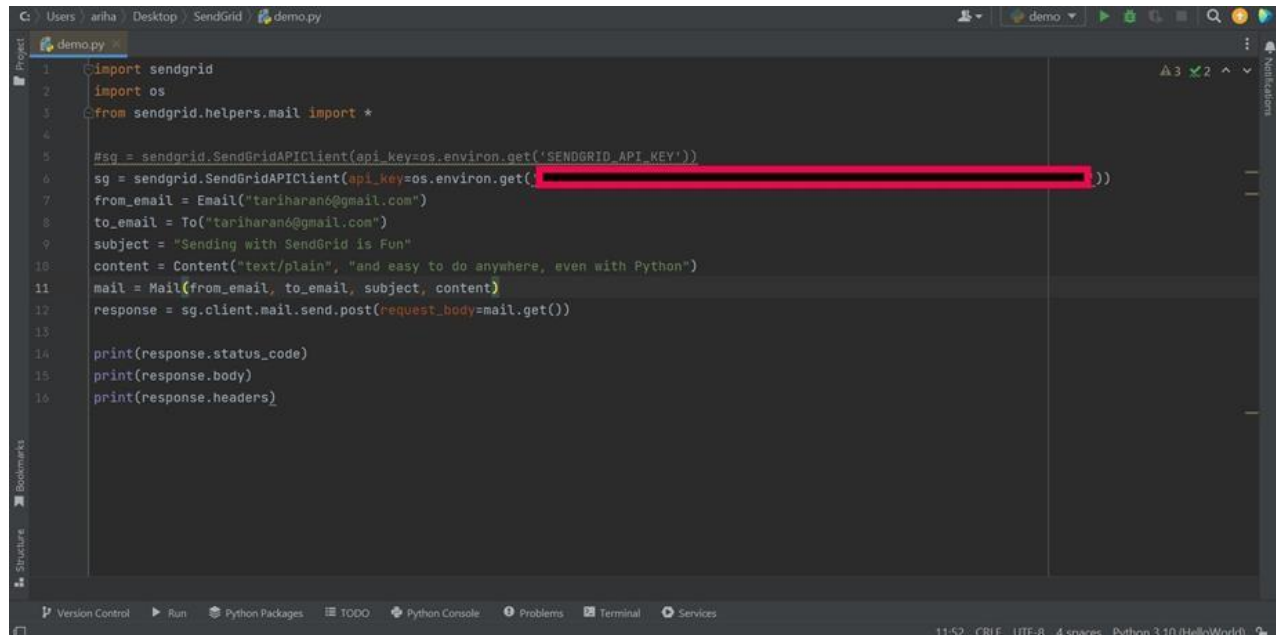**Python 2.6, 2.7, 3.4 or 3.5.**

STEP 2:

Create an API key

STEP 3:

INSTALL
PAKAGE: > pip install sendgrid

SETP 4:

SEND EMAIL

SENDGRID PYTHON CODE :

```python
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
   Python</strong>')
10 try:
11     sg = SendGridAPIClient(os.environ.get('SENDGRID_API_KEY'))
12     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:

```python
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
```

```python
13    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
23     def __init__(self, response):
24         """
25         :param response: The return value from a open call
26                          on a urllib.build_opener()
27         :type response:  urllib response object
28         """
29         self._status_code = response.getcode()
30         self._body = response.read()
31         self._headers = response.info()
32
33     @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
```

```python
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 []
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),
    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
```

```python
        parameters
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):
```

```python
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. This is
     separated into
164             it's own function, so we can mock it easily for testing.
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}
```

```python
{body}'.format(
                method=request.get_method(),
                status=exc.status_code,
                body=exc.body))
            raise exc

    def _(self, name):
        """Add variable values to the url.
            (e.g. /your/api/{variable_value}/call)
            Another example: if you have a Python reserved word,
    such as global,
            in your url, you must use this method.

        :param name: Name of the url segment
        :type name: string
        :return: Client object
        """
        return self._build_client(name)

    def __getattr__(self, name):
        """Dynamically add method calls to the url, then call a
    method.
            (e.g. client.name.name.method())
            You can also add a version number by using
    .version(<int>)

        :param name: Name of the url segment or method call
        :type name: string or integer if name == version
        :return: mixed
        """
        if name == 'version':
            def get_version(*args, **kwargs):
                """
                    :param args: dict of settings
                    :param kwargs: unused
```

```python
212                     :return: string, version
213                     """
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                 :param timeout: HTTP request timeout. Will be
   propagated to
230                     urllib client
231                 :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:
```

```python
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
```

```
276                        response = Response(
277                             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):
```