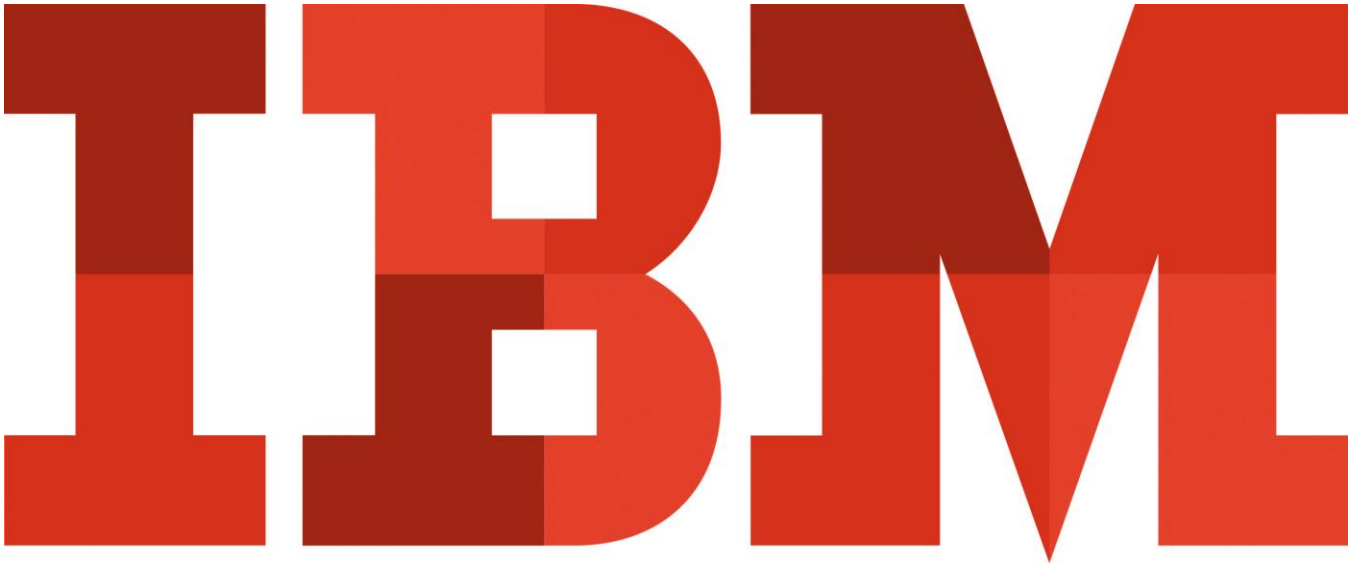
Python, Watson and IBM Bluemix

A guide for hackathons

Lennart Frantzell alf@us.ibm.com
2/5/2017



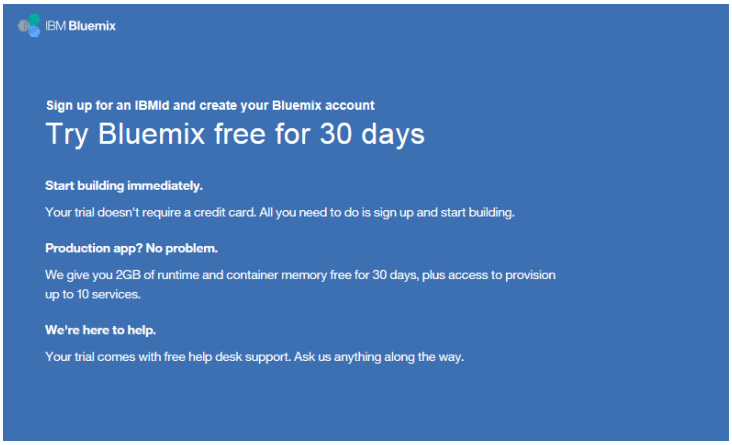
A digital copy of this lab and code snippets can be found at:
`http://ibm.biz/<<TODO>>`

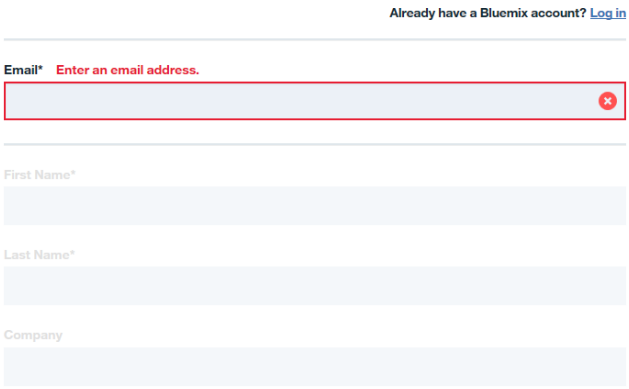
Quick Introduction

When I attend Hackathons I am often asked if the IBM Bluemix cloud supports Python and how you can combine Python with Watson and other services on the IBM Bluemix cloud. This little guide aims to provide a brief introduction to Python on Bluemix and show how you can use the Watson Personality Insights cognitive service with Python https://www.ibm.com/watson/developercloud/personality-insights.html. Enjoy.

Getting started

1. To work with Python on Bluemix, let's begin by signing up to a 30 day free Bluemix account at **http://bluemix.net**



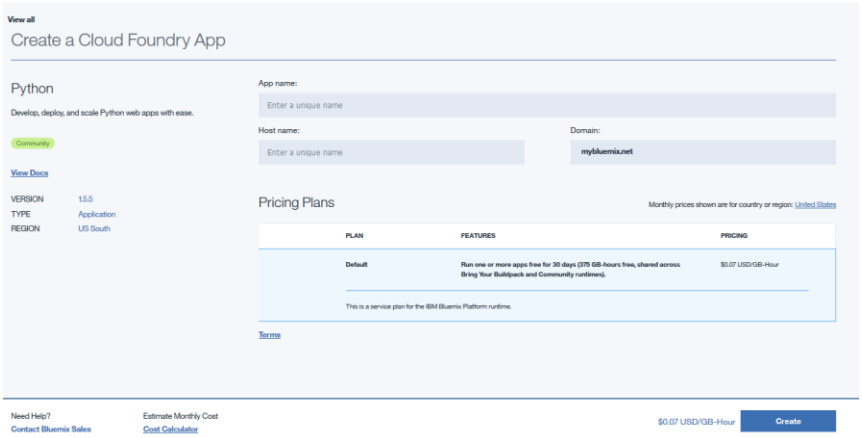


Once we have signed up to Bluemix we need to take the following preparatory steps:

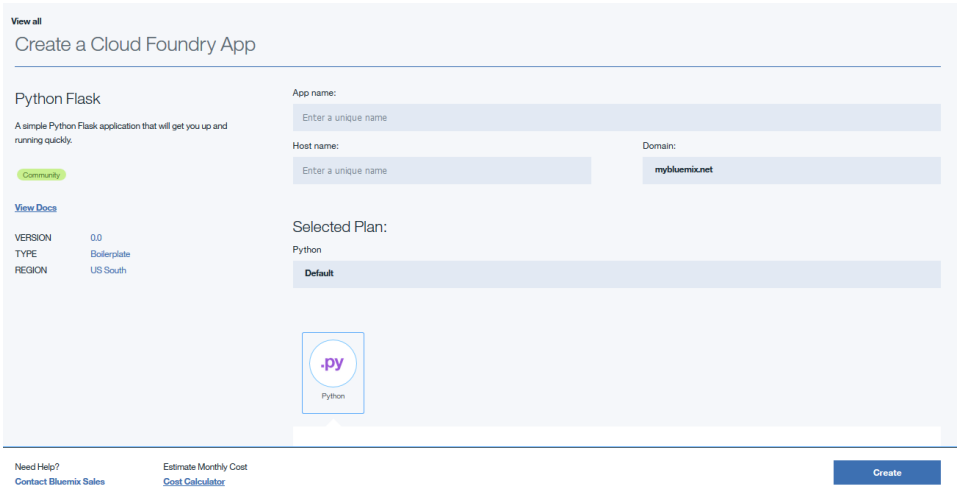
1. Download the **Git command** for our operating system: **https://git-scm.com/downloads**. We will use Git to pull the Watson code from the GitHub repository where it is hosted down to our laptop.
2. Download the **Cloudfoundry Command Line Interface (CLI)** for our operating system: **https://github.com/cloudfoundry/cli/releases.** We will use the Cloudfoundry cf command to communicate with Bluemix from the command line on our laptops.
3. Download the Watson Developer Cloud Python SDK **https://github.com/watson-developer-cloud/python-sdk** with this command: **pip install --upgrade watson-developer-cloud.** It has been tested on Python from 2.7 to 3.6**.** This SDK allows us to write Python apps that connect to many of the Watson Cognitive Services
4. And if you don't have Python installed you can do so from this website: **https://www.python.org/downloads/**

Now we are ready to begin and we will find **Python** https://console.ng.bluemix.net/catalog/starters/python/ and **Python Flask** https://console.ng.bluemix.net/catalog/starters/python-flask/ listed in the Bluemix catalog**.**

We will only familiarize ourselves with these two build packs at this moment but not yet launch them.
Later, when we are ready to launch them, we just have to click on the **create button** to get them running in the Bluemix cloud



https://console.ng.bluemix.net/catalog/starters/python/?taxonomyNavigation=apps



https://console.ng.bluemix.net/catalog/starters/python-flask/?taxonomyNavigation=apps

More information on Python on Bluemix can be found on this link:
https://console.ng.bluemix.net/docs/runtimes/python/index.html#python_runtime

To use the Watson services with Python we will use the Watson Service APIs, most of which come with Python support:

- Alchemy Language: https://www.ibm.com/watson/developercloud/alchemy-language/api/v1/
- Conversation: https://www.ibm.com/watson/developercloud/conversation/api/v1/
- Document Conversion: https://www.ibm.com/watson/developercloud/document-conversion/api/v1/
- Language Translator: https://www.ibm.com/watson/developercloud/language-translator/api/v2/
- Natural Language Classifier: https://www.ibm.com/watson/developercloud/natural-language-classifier/api/v1/
- Personality Insights: https://www.ibm.com/watson/developercloud/personality-insights/api/v3/
- Retrieve and Rank: No Python API support
- Tone Analyzer: https://www.ibm.com/watson/developercloud/tone-analyzer/api/v3/
- Speech to Text, no Python API Support
- Text to Speech: no Python API Support
- Project Into: no Python API Support
- Visual Recognition: https://www.ibm.com/watson/developercloud/visual-recognition/api/v3/
- Alchemy Data News: https://www.ibm.com/watson/developercloud/alchemydata-news/api/v1/
- Discovery Service: https://www.ibm.com/watson/developercloud/discovery/api/v1/
- Tradeoff Analytics: https://www.ibm.com/watson/developercloud/tradeoff-analytics/api/v1/

A list of Python sample code with Watson Services can be found at this link: https://github.com/watson-developer-cloud/python-sdk/tree/master/examples

Let’s download the Watson Personality Insights starter application repo from GitHub

We are now ready to get to work.

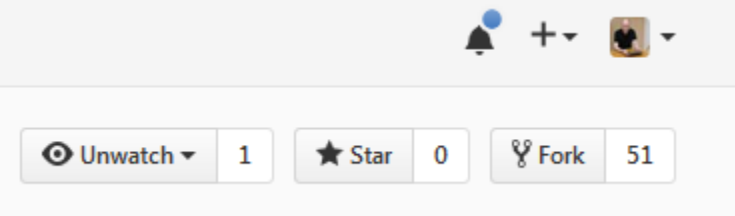
The Watson Personality Insights service https://www.ibm.com/watson/developercloud/personality-insights.html is one of the most popular Watson cognitive services at hackathons and beyond.

A **Personality Insights Python Starter Application** can be found on https://github.com/watson-developer-cloud/personality-insights-

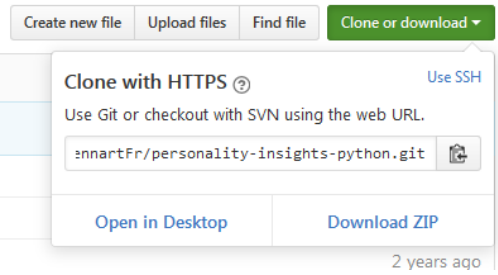
python. This is client code that allows us to write Python apps that communicate the the Watson Persoanlity Insights service running on Bluemix.

And Please Note that the Personality Insights Python Starter Application calls the Watson Developer Cloud Python SDK that we did a pip install on earlier in this workshop.

We will start by doing a git fork of the original **Personality Insights Python Starter Application** GitHub repository to our own GitHub repository. You do that in the GitHub UI in the personalityinsights-python repo by clicking on the **Fork** button as shown below:



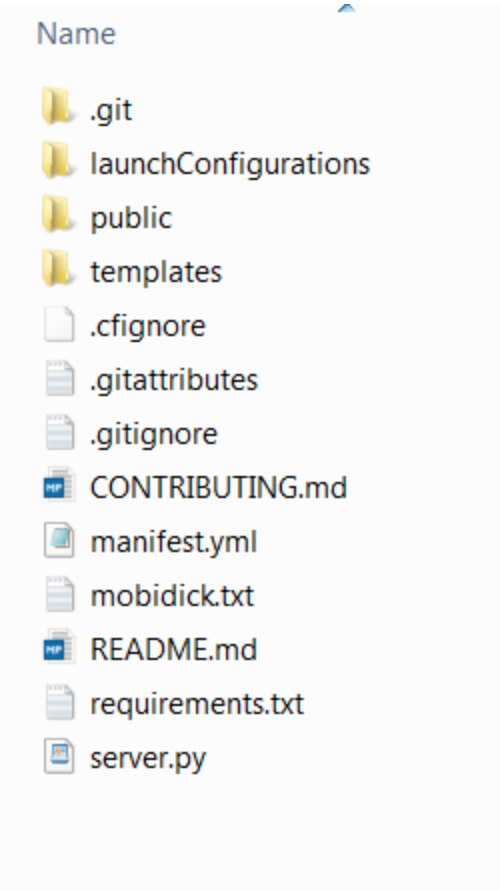
This will create a copy of the official repo in your own GitHub directory.
You can then do a git clone of the forked repository to your own drive on your laptop by first clicking on the green "Clode of download" button as shown below. This will show the new url to your repo in your own directory. Make certain that the URL points to your own GitHub directory.



Then, from the console window on your own laptop, issue the git clone command to copy down your own version of the repo.
In this case I am copying down my own copy of the repo. Your copy will obviously look different.

git clone https://github.com/LennartFr/personality-insights-python.git to dowload the code from your forked repo**.**

And the result will be the following files and directories:



Now let's look at some of the key files in our project. The first one is the **manifest.yml** file.

```
---
declared-services:
  personality-insights-service:
    label: personality_insights
    plan: tiered

applications:
- name: personality-insights-python
  command: python server.py
  path: .
  memory: 256M
  services:
  - personality-insights-service
```

The **manifest.yml** file contains the meta information needed by Bluemix to cofigure and launch our app. For more on this file see: https://docs.cloudfoundry.org/devguide/deploy-apps/manifest.html. YML http://yaml.org/ is a human-readable data serialization language, commonly used for configuration files.

The most important entries in this file is the declared service which is **personality-insights-service**. Another is the application name, which is **personality-insights-python** . And the amount of memory which is 256M.

We will soon do a **cf push** command to Bluemix to create the application that is currently named as **personality-insights**-python in the YML file. But first we need to change the name in the YML file to something unique.

Also, when we do the push command we will create the app in Bluemix. But we will not create the **personality-insights-service** during the push. To create the personality-insights-service we run the following cf command from the command line. Let's do so now:

```
E:\Mina Projekt\\personality-insights-python>cf create-service personality_insights tiered personality_insights_service
Creating service instance personality_insights_service in org alf@us.ibm.com / space dev as alf@us.ibm.com...
OK

Attention: The plan `tiered` of service `personality_insights` is not free.  The instance `personality_insights_service1` will incur a cost.  Contact your administrator if you think this is in error.
```

"Tiered" is the type of service available. See for more information: https://console.ng.bluemix.net/catalog/services/personality-insights/

And please notice that generic type name of the service is **personality_insights** and the name of our instance of the service is **personality_insights_service**

Now when we do a **cf services** command we will see the new service we created.

personality-insights-service	personality_insights	tiered	personality-insights-python	update succeeded

It is good to familiarize oneself with the **manifest.yml file** and all the options available in it**.**

Let's also familiarize ourselves with the **requirements.txt** which contains some producrs that are required to run the app:

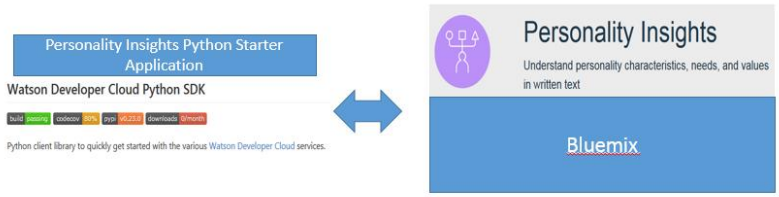
```
cherrypy
mako
requests
```

We may have to install some of these when we run the app on our laptop.

And finally of course the **server.py** file is the executable file.

So let's see where we are now.

1. We have installed the Personality Insights cognitive service in Bluemix
2. We have done a pip install of the Watson Developer Cloud Python SDK
3. We have downloaded the Personality Insights Python Starter Application



We are now ready to push our app to Bluemix.

Let's push the app to Bluemix

We are now ready to do the cf push command, wich will read the manifest.yml file and upload our app to Bluemix.

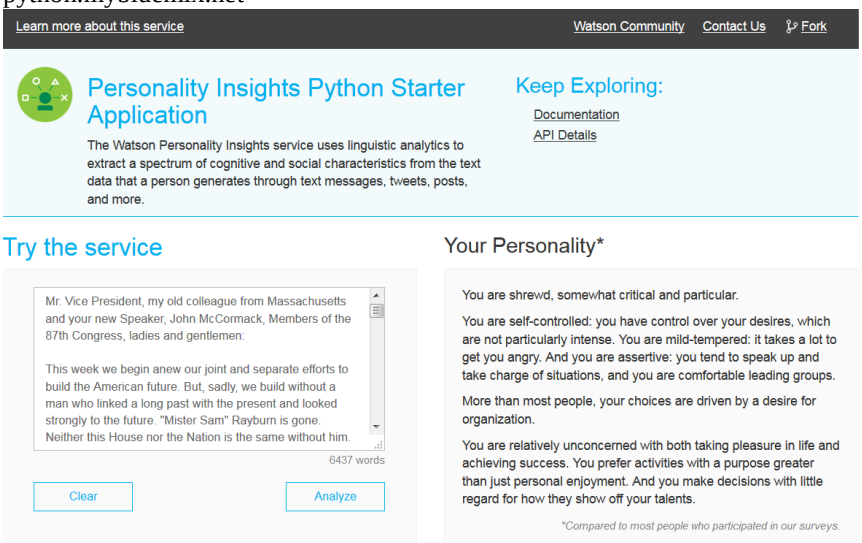
```
App personality-insights-python was started using this command `python server.py`

Showing health and status for app personality-insights-python in org alf@us.ibm.com / space dev as alf@us.ibm.com...
OK
```

```
requested state: started
instances: 1/1
usage: 256M x 1 instances
urls: personality-insights-python.mybluemix.net
last uploaded: Fri Feb 3 17:13:51 UTC 2017
stack: cflinuxfs2
buildpack: python 1.5.5

     state     since                  cpu    memory         disk          details
#0   running   2017-02-03 09:15:52 AM   0.0%   23.6M of 256M   119.1M of 1G
```

The app is now running on Bluemix. If we go to the URL shown above we will see it: http://personality-insights-python.mybluemix.net



And now let's run the app on our laptop

Important note:

In order to run our app locally on our laptop we need to configure the app to recognize the credentials from the service running on Bluemix that we just created. To get those credentials we do a **cf env** command for our app on our laptop like this**: cf env personality-insights-python.**

```
System-Provided:
{
 "VCAP_SERVICES": {
  "personality_insights": [
   {
    "credentials": {
     "password": "kClpmximxxxx",
     "url": "https://gateway.watsonplatform.net/personality-insights/api",
     "username": "971b73b4-abaa-4429-9c76-ec803f93xxxx"
    },
    "label": "personality_insights",
    "name": "personality-insights-service",
    "plan": "tiered",
    "provider": null,
    "syslog_drain_url": null,
    "tags": [
     "watson",
```

```
    "ibm_created",
    "ibm_dedicated_public"
   ]
  }
 ]
}
}
```

We need to insert the **userid, password** and **url** from the VCAP Services into the **server.py** file on our laptop as shown below:

```
class PersonalityInsightsService:
  """Wrapper on the Personality Insights service"""

  def __init__(self, vcapServices):
    """
    Construct an instance. Fetches service parameters from VCAP_SERVICES
    runtime variable for Bluemix, or it defaults to local URLs.
    """

    # Local variables
```

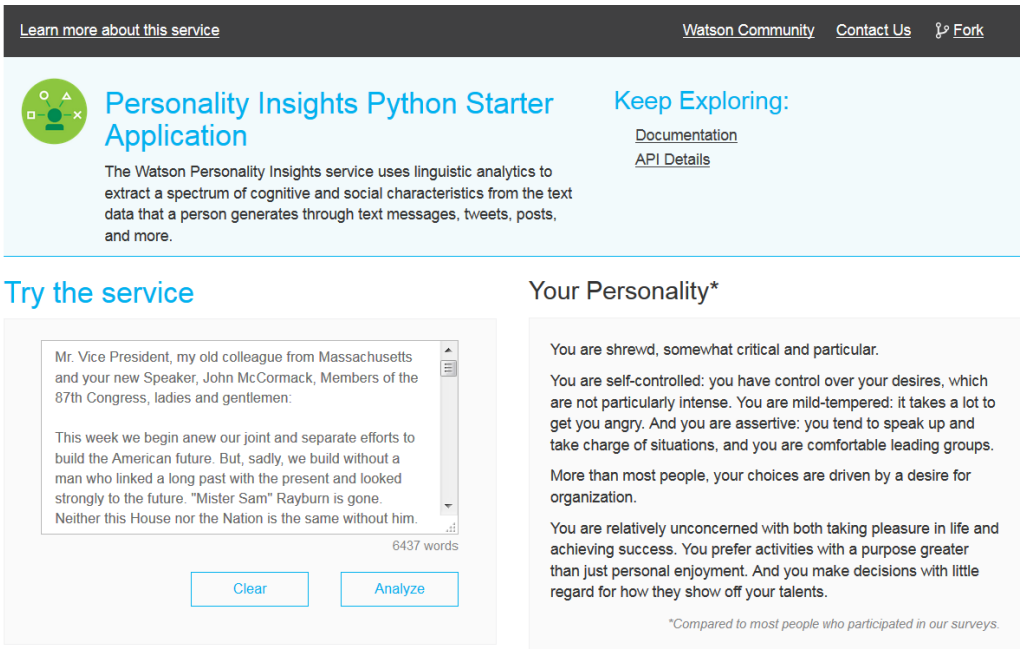
```
      self.url = "https://gateway.watsonplatform.net/personality-insights/api"
      self.username = "6da14925-e64a-447b-a3cb-a2e3e4d6xxxx"
      self.password = "03WsTqyvxxxx"
```

We are now ready to run our Python app on our local laptop drive by doing: **python server.py**
When we do so we may be asked to install cherrypy and mako, in case we don't already have them installed. Which we do with pip https://pypi.python.org/pypi/pip, Pythons install manager.

```
E:\Mina Projekt\testgitclone\gitclone\personality-insights-python1\personality-insights-python>python server.py
Listening on 127.0.0.1:3000
[03/Feb/2017:10:06:09] ENGINE Listening for SIGTERM.
[03/Feb/2017:10:06:09] ENGINE Bus STARTING
[03/Feb/2017:10:06:09] ENGINE Set handler for console events.
[03/Feb/2017:10:06:09] ENGINE Started monitor thread 'Autoreloader'.
[03/Feb/2017:10:06:09] ENGINE Started monitor thread '_TimeoutMonitor'.
[03/Feb/2017:10:06:10] ENGINE Serving on http://127.0.0.1:3000
[03/Feb/2017:10:06:10] ENGINE Bus STARTED
```

We are now up and running on or laptop on IP address

You should now see the same GUI as when the app is ruinning in Bluemix. And we can analyze personalities just as we can when our app is running on Bluemix.



We can now begin to modify the app on our laptop and create a new version that we push up to Bluemix. Please note that when we run the app locally on our laptop we still connect to the Watson Personaly Insights service running on Bluemix.

Let's now strip the app down to the bare minimum

And please note that if we want to we can create a Python app on our laptop that does not run at all on Bluemix, but only connects to the Watson cignitive service running on Bluemix. Let's do so now.

Let's strip our app down to bare essentials. The version below should run on your laptop without problem. Please notice how it connects to the Watson Service on Bluemix.

```
import os
import json
from jinja2 import Template
import requests
from flask import Flask, render_template, request, url_for

url = "https://gateway.watsonplatform.net/personality-insights/api"
username = "6da14925-e64a-447b-a3cb-a2e3e4d6xxxx"
password = "03WsTqyvxxxx"

text=("From the moment that the French defenses at Sedan and on the Meuse were broken at the end of the second week of May,"
    " only a rapid retreat to Amiens and the south could have saved the British and French Armies who had entered Belgium"
            " at the appeal of the Belgian King; but this strategic fact was not immediately realized."
            " The French High Command hoped they would be able to close the gap, and the Armies of the north were under their orders."
            " Moreover, a retirement of this kind would have involved almost certainly the destruction of the fine Belgian Army of over"
            " 20 divisions and the abandonment of the whole of Belgium. Therefore, when the force and scope of the German penetration were"
            " realized and when a new French Generalissimo, General Weygand, assumed command in place of General Gamelin, an
```

```
effort was made"
            " by the French and British Armies in Belgium to keep on holding the right hand of the Belgians and to give their own
right hand"
            " to a newly created French Army which was to have advanced across the Somme in great strength to grasp it.")


response = requests.post(url + "/v2/profile",
   auth=(username, password),
   headers = {"content-type": "text/plain"},
   data=text
)

try:

   print (json.loads(response.text))
except:
   raise Exception("Error processing the request, HTTP: %d" % response.status_code)
```

Invoking this stripped app on our laptop gives us this output:

```
E:\test\personality-insights-python>python laptopapp.py
{'tree': {'id': 'r', 'name': 'root', 'children': [{'id': 'personality', 'name': 'Big 5', 'children': [{'category': 'personality', 'id':
'Openness_parent', 'name': 'Openness', 'children': [{'name': 'Openness', 'children': [{'category': 'personality', 'id': 'Adventurousness',
'name': 'Adventurousness
', 'sampling_error': 0.0539097484, 'percentage': 0.4757172877125303}, {'category': 'personality', 'id': 'Artistic interests', 'name':
'Artistic interests', 'sampling_error': 0.1097414144, 'percentage': 0.9679908476392554}, {'category': 'personality', 'id':
'Emotionality', 'name': 'Emotionality', 'sa
pling_error': 0.050441622400000004, 'percentage': 0.5872606987877054}, {'category': 'personality', 'id': 'Imagination', 'name':
'Imagination', 'sampling_error': 0.0680641844, 'percentage': 0.7304094200245063}, {'category': 'personality', 'id': 'Intellect', 'name':
'Intellect', 'sampling_error': 0.0
97492336, 'percentage': 0.9985641463714839}, {'category': 'personality', 'id': 'Liberalism', 'name': 'Authority-challenging',
'sampling_error': 0.08779632679999999, 'percentage': 0.9987290906532491}], 'category': 'personality', 'id': 'Openness',
'sampling_error': 0.0645750316, 'percentage': 0.99879
8993975366}, {'name': 'Conscientiousness', 'children': [{'category': 'personality', 'id': 'Achievement striving', 'name': 'Achievement
striving', 'sampling_error': 0.1038650008, 'percentage': 0.7778965474353808}, {'category': 'personality', 'id': 'Cautiousness', 'name':
'Cautiousness', 'sampling_er
or': 0.0961618044, 'percentage': 0.8971738462214329}, {'category': 'personality', 'id': 'Dutifulness', 'name': 'Dutifulness',
'sampling_error': 0.064850272, 'percentage': 0.7591579992991817}, {'category': 'personality', 'id': 'Orderliness', 'name':
'Orderliness', 'sampling_error': 0.0739024068, 'pe
rcentage': 0.14120868450178892}, {'category': 'personality', 'id': 'Self-discipline', 'name': 'Self-discipline', 'sampling_error':
0.0494612016, 'percentage': 0.10539466496402566}, {'category': 'personality', 'id': 'Self-efficacy', 'name': 'Self-efficacy',
'sampling_error': 0.0971073876, 'percentage
': 0.7245635717567557}], 'category': 'personality', 'id': 'Conscientiousness', 'sampling_error': 0.08064647520000001, 'percentage':
0.5776000093587051}, {'name': 'Extraversion', 'children': [{'category': 'personality', 'id': 'Activity level', 'name': 'Activity level',
'sampling_error': 0.0824344144,
'percentage': 0.6532898525030351}, {'category': 'personality', 'id': 'Assertiveness', 'name': 'Assertiveness', 'sampling_error':
0.0877454512, 'percentage': 0.5729149805473628}, {'category': 'personality', 'id': 'Cheerfulness', 'name': 'Cheerfulness',
'sampling_error': 0.1103904608, 'percentage': 0
006174126673610447}, {'category': 'personality', 'id': 'Excitement-seeking', 'name': 'Excitement-seeking', 'sampling_error':
0.08424550080000001, 'percentage': 0.18850510604321014}, {'category': 'personality', 'id': 'Friendliness', 'name': 'Outgoing',
'sampling_error': 0.0792927336, 'percentage': 0
03951013686121535}, {'category': 'personality', 'id': 'Gregariousness', 'name': 'Gregariousness', 'sampling_error':
0.060546519599999994, 'percentage': 0.007834403339712415}], 'category': 'personality', 'id': 'Extraversion', 'sampling_error':
0.060327493600000004, 'percentage': 0.1051739589042871},
{'name': 'Agreeableness', 'children': [{'category': 'personality', 'id': 'Altruism', 'name': 'Altruism', 'sampling_error': 0.0747597584,
'percentage': 0.8126695466976206}, {'category': 'personality', 'id': 'Cooperation', 'name': 'Cooperation', 'sampling_error':
0.0834654396, 'percentage': 0.6771718
59602147}, {'category': 'personality', 'id': 'Modesty', 'name': 'Modesty', 'sampling_error': 0.0596197876, 'percentage':
0.3145221820771268}, {'category': 'personality', 'id': 'Morality', 'name': 'Uncompromising', 'sampling_error': 0.066417394,
```

'percentage': 0.3468313696696078}, {'category': 'perso
ality', 'id': 'Sympathy', 'name': 'Sympathy', 'sampling_error': 0.1024640644, 'percentage': 0.9939371896101595}, {'category':
'personality', 'id': 'Trust', 'name': 'Trust', 'sampling_error': 0.0604120784, 'percentage': 0.9722486467505242}], 'category':
'personality', 'id': 'Agreeableness', 'samplin
_error': 0.10125405, 'percentage': 0.007412874064475772}, {'name': 'Emotional range', 'children': [{'category': 'personality', 'id':
'Anger', 'name': 'Fiery', 'sampling_error': 0.0985949952, 'percentage': 0.554783230873663}, {'category': 'personality', 'id': 'Anxiety',
'name': 'Prone to worry', 'sa
pling_error': 0.0583679468, 'percentage': 0.592202679424692}, {'category': 'personality', 'id': 'Depression', 'name': 'Melancholy',
'sampling_error': 0.0625670476, 'percentage': 0.9167665686355615}, {'category': 'personality', 'id': 'Immoderation', 'name':
'Immoderation', 'sampling_error': 0.056169
76, 'percentage': 0.6191321068676958}, {'category': 'personality', 'id': 'Self-consciousness', 'name': 'Self-consciousness',
'sampling_error': 0.060505887200000004, 'percentage': 0.9010980518035747}, {'category': 'personality', 'id': 'Vulnerability',
'name': 'Susceptible to stress', 'sampling_error
': 0.09014683999999999, 'percentage': 0.4590151036667137}], 'category': 'personality', 'id': 'Neuroticism', 'sampling_error':
0.095721062, 'percentage': 0.7260945739455659}], 'percentage': 0.9987998993975366}]}, {'id': 'needs', 'name': 'Needs', 'children':
[{'category': 'needs', 'id': 'Liberty_paren
', 'name': 'Liberty', 'children': [{'category': 'needs', 'id': 'Challenge', 'name': 'Challenge', 'sampling_error': 0.087061484, 'percentage':
0.02955560220212583}, {'category': 'needs', 'id': 'Closeness', 'name': 'Closeness', 'sampling_error': 0.0859332952, 'percentage':
0.0262450435966487}, {'cate
ory': 'needs', 'id': 'Curiosity', 'name': 'Curiosity', 'sampling_error': 0.1243616188, 'percentage': 0.39030178416497313}, {'category':
'needs', 'id': 'Excitement', 'name': 'Excitement', 'sampling_error': 0.1135503056, 'percentage': 0.04358807383818103}, {'category':
'needs', 'id': 'Harmony', 'name
': 'Harmony', 'sampling_error': 0.113754904, 'percentage': 0.00944920261173171}, {'category': 'needs', 'id': 'Ideal', 'name': 'Ideal',
'sampling_error': 0.1031497164, 'percentage': 0.030451937304600996}, {'category': 'needs', 'id': 'Liberty', 'name': 'Liberty',
'sampling_error': 0.1501821348, 'perce
tage': 0.002051373166778392}, {'category': 'needs', 'id': 'Love', 'name': 'Love', 'sampling_error': 0.1046808284, 'percentage':
0.004900993820087518}, {'category': 'needs', 'id': 'Practicality', 'name': 'Practicality', 'sampling_error': 0.0909585948, 'percentage':
0.004024173522066021}, {'category'
'needs', 'id': 'Self-expression', 'name': 'Self-expression', 'sampling_error': 0.0846391764, 'percentage': 0.005101295668730243 5},
{'category': 'needs', 'id': 'Stability', 'name': 'Stability', 'sampling_error': 0.11063459760000001, 'percentage':
0.04135023623323253}, {'category': 'needs', 'id': 'S
ructure', 'name': 'Structure', 'sampling_error': 0.0831281048, 'percentage': 0.6024223462382614}], 'percentage':
0.002051373166778392}]}, {'id': 'values', 'name': 'Values', 'children': [{'category': 'values', 'id': 'Hedonism_parent', 'name':
'Hedonism', 'children': [{'category': 'values', 'id': 'Co
servation', 'name': 'Conservation', 'sampling_error': 0.0706480588, 'percentage': 0.07328857952782603}, {'category': 'values', 'id':
'Openness to change', 'name': 'Openness to change', 'sampling_error': 0.06778235, 'percentage': 0.3330974091577948}, {'category':
'values', 'id': 'Hedonism', 'name':
Hedonism', 'sampling_error': 0.14212298439999999, 'percentage': 0.007424223711857192}, {'category': 'values', 'id': 'Self-
enhancement', 'name': 'Self-enhancement', 'sampling_error': 0.10781349159999999, 'percentage': 0.09418805332865399},
{'category': 'values', 'id': 'Self-transcendence', 'name': '
elf-transcendence', 'sampling_error': 0.08574562599999999, 'percentage': 0.03830133136347541}], 'percentage':
0.007424223711857192}]}]}, 'source': '*UNKNOWN*', 'word_count_message': 'There were 188 words in the input. We need a
minimum of 600, preferably 1,200 or more, to compute statistically sign
ficant estimates', 'id': '*UNKNOWN*', 'processed_lang': 'en', 'word_count': 188, 'warnings': [{'message': 'There were 188 words in
the input. We need a minimum of 600, preferably 1,200 or more, to compute statistically significant estimates', 'warning_id':
'WORD_COUNT_MESSAGE'}]}

We can now build upp our own app from this minimal version, leveraging the Watson Congitive services on Bluemix.

Wrapping up and Git commands

We leave it as an exercise to modify the application. When ready, use the following Git commands:

- **git status** to see changes

- **git add filename** to add any untracked files you might have added
- **git commit -m "message"** to commit the changes
- **git pull** to pull down any changes you may have made on JazzHub
- **git push origin master** to push back to the git repo

And as we make changes to out code we update our GitHub repository.

We can use the same method to develop apps with the other Watson Cognitive Services listed above..

This concludes this lab.