The folder structure of the dataQualityFramework is as shown below

Prerequisites:

C:/data/dataQualityFramework/config/file\_config.csv(HEADER: data\_file\_path,expectation\_file\_path)

C:/data/dataQualityFramework/output/

C:/data/dataQualityFramework/framework.ipynb

C:/data/input\_files/….



* Data: This folder contains the data csv files (There is no hard and fast rule that the data files should be under this folder)
* Config: This folder is where the file\_config.csv file resides and where entries of the data file and the expectations file needs to be made
* Expectations: This folder contains all the expectation json files (There is no hard and fast rule that the data files should be under this folder)
* Output : This folder is where the output validation results will be stored.

Steps to add a new expectation and data file:

1. Choose any data csv file of your liking which has a header and place it anywhere in your PC and copy the file path of it
2. Go to dataQualityFramework -> config -> open the file\_config.csv file and add a new entry under the data\_file\_path column with the path copied in step 1
3. Create a .json file with a name of your liking and add the following template

**{**

**"data\_asset\_type": "Dataset",**

**"expectation\_suite\_name": "default",**

**"expectations": [**

**{**

**"expectation\_type": "<expectation type you want to use>",**

**"kwargs": {**

**"column": "<column name>"**

**},**

**"meta": {}**

**},**

**],**

**"ge\_cloud\_id": null,**

**"meta": {**

**"great\_expectations\_version": "0.15.49"**

**}**

**}**

4. replace the place holder values at the above snippet with the expectations and columns of your choice. The list of expectations can be found here <https://greatexpectations.io/expectations/> . Copy the expectation type name from the expectation of your choice from the gallery link above and replace it in the .json file in the placeholder. To add any additional **Kwargs** (see line 10 in above snippet) the expectation has just add the argument below the **column** argument

5. In order to add more expectations just copy the lines from 8 to 14 in the above snippet and change the placeholders with the expectation name and its other properties

6. after finishing the expectations list ,save the .json file and copy the path of the file

7. Copy the path name to under the column expectation\_file\_path in the fiile opened in step 2

8. Now open the framework.ipynb notebook and run all the cells

9. After successful run, you can check your results in the validation\_result.csv file under the output folder.

import great\_expectations as ge

from csv import reader,writer

import os

import datetime

def read\_csv\_file(file\_path : str):

return ge.read\_csv(file\_path)

def validate\_expectations(data\_frame,expectation\_file\_path):

return data\_frame.validate(expectation\_suite= expectation\_file\_path)

HEADER=["FILE\_NAME","COLUMN\_NAME","TEST\_TYPE","TEST\_STATUS","TOTAL\_EVALUATIONS","EVALUATIONS\_PASSED","EVALUATIONS\_FAILED","SUCCESS\_RATIO","VALIDATION\_TIME","CURRENT\_TIME"]

# skip first line i.e. read header first and then iterate over each row od csv as a list

with open('./config/file\_config.csv', 'r') as read\_obj:

csv\_file = open("./output/validation\_result.csv", "w")

csv\_file.write(",".join(HEADER))

csv\_file.write("\n")

csv\_reader = reader(read\_obj)

header = next(csv\_reader)

# Check file as empty

if header != None:

# Iterate over each row after the header in the csv

for file\_path,expectation\_file\_path in csv\_reader:

# row variable is a list that represents a row in csv

file\_name=os.path.basename(file\_path)

current\_time = datetime.datetime.now().strftime("%d/%m/%Y %H:%M:%S")

df = read\_csv\_file(file\_path)

result = validate\_expectations(df,expectation\_file\_path)

total\_evaluations=str(result["statistics"]["evaluated\_expectations"])

evaluations\_passed=str(result["statistics"]["successful\_expectations"])

evaluations\_failed=str(result["statistics"]["unsuccessful\_expectations"])

success\_ratio=str(result["statistics"]["success\_percent"])

validation\_time=result["meta"]["validation\_time"]

for test in result["results"]:

test\_type=test["expectation\_config"]["expectation\_type"]

test\_status=str(test["success"])

column\_name=test["expectation\_config"]["kwargs"]["column"]

line = [file\_name,column\_name,test\_type,test\_status,total\_evaluations,evaluations\_passed,evaluations\_failed,success\_ratio,validation\_time,current\_time]

csv\_file.write(','.join(line))

csv\_file.write('\n')

csv\_file.close()