User Acceptance Test Script: User Story 2

(Last updated 4-13-2021)

1. The user will begin by accessing the GitHub repository that contains all Weather Eyes project related files and ensuring the files are downloaded and accessible in a common directory. (Please refer to the Deployment document for a list of all programs that will need to be collected and available in the same folder).
2. The process of scraping forecast data for weather.gov, weather.com, and accuweather.com is initiated by running the WeatherEyesMain.py program.
3. When WeatherEyesMain.py is running, hourly and extended daily forecast data is scraped once per hour for every successive hour after the program continues to run.
   1. Note, in some instances, we have learned that the NatWeatherServ scrape will encounter a 500 error. In cases where this occurs, the function may need to be re-run.
4. The MongoDB client where all of the scraped forecast data is available is accessible at: "mongodb+srv://team:team@cluster0.nmfhg.mongodb.net/myFirstDatabase?retryWrites=true&w=majority"
5. For the purposes of navigating the data, the MongoDB data is mapped in the following way:
   1. Cluster name = “Cluster0”
   2. Database1 = “AccuWeather”
      1. Collection1 = “AccuWeather\_hourly”
      2. Collection2 = “AccuWeather\_tenday”
   3. Database2 = “NatWeather”
      1. Collection1 = “NatWeather\_hourly”
      2. Collection2 = “NatWeather\_sevenday”
   4. Database3 = “WeatherCom”
      1. Collection1 = “WeatherCom\_hourly”
      2. Collection2 = “WeatherCom\_tenday”
6. All data collections for each database—hourly, sevenday, and tenday—will be captured hourly on the hour. The starting time for data collection is dependent upon when the WeatherEyesMain.py program is started.
7. Within each collection, the following data dictionaries can be used to query information using pymongo. The examples below are for an April 12, 2021 forecast that was collected for the AccuWeather database. This example can be used as a template for queries using the MongoDB db.collection.find() method:

AccuWeather\_tenday  
"date" : "Apr122021","PrecipitationProb" : "0%","Max Temp (in F)" : "65","Min Temp (in F)" : "49","Description" : "Times of sun and clouds","Time Collected" : "Apr122021 2PM”

AccuWeather\_hourly  
"date" : "Apr122021",  
“time”: “10PM”"PrecipitationProb" : "37%","Temperature (in F)" : "50",  
"Description" : "Mostly Cloudy","Time Collected" : "Apr112021 2PM”

1. Note regarding the data dictionaries for all databases and collections:
   1. For the extended forecasts (tenday, sevenday), the same dictionary elements are available for NatWeather and WeatherCom as the sample elements for AccuWeather tenday displayed above.
   2. For the hourly forecasts, the same dictionary elements are available for NatWeather and WeatherCom as the sample elements for AccuWeather tenday displayed above.
2. The user will access the data directly by querying the MongoDB database using the pymongo package within a python and the MongoDB db.collection.find() method.
3. Since the forecast data is dynamic, the user can compare the results of a query on the data to the actual website.
   1. If database1 is used, a real-time visual comparison can be made by doing to weather.gov and entering the Pittsburgh zip code 15282.
   2. If database2 is used, a real-time visual comparison can be made by doing to weather.com and entering the Pittsburgh zip code 15282.
   3. If database3 is used, a real-time visual comparison can be made by doing to accuweather.com and entering the Pittsburgh zip code 15282.