Technical details for web scraping power outage data used in "Inside the Black Box: Unequal Infrastructure and Institutional Bias in Energy Access"

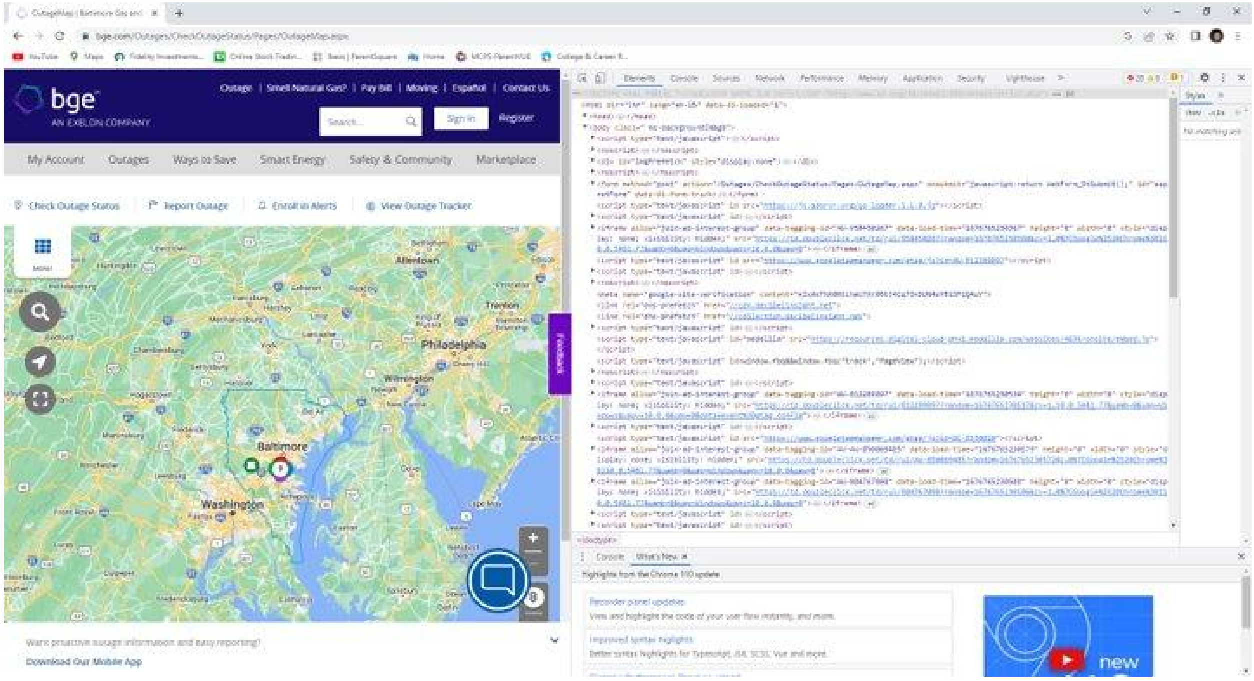
Feb, 2023
Nathan Shan, Montgomery Blair High School Maryland

1. Web Scraping with Python Requests, Beautiful Soup, and Selenium

   Web scraping is the process of extracting usable data from different webpages. In this project, I scraped text from utility outage maps by loading a URL and loading the HTML, CSS and JavaScript code.

2. Python Requests: used to send HTTP/1.1 requests to load URLs

The screenshot shows the front page on the left and HTML code on the right.

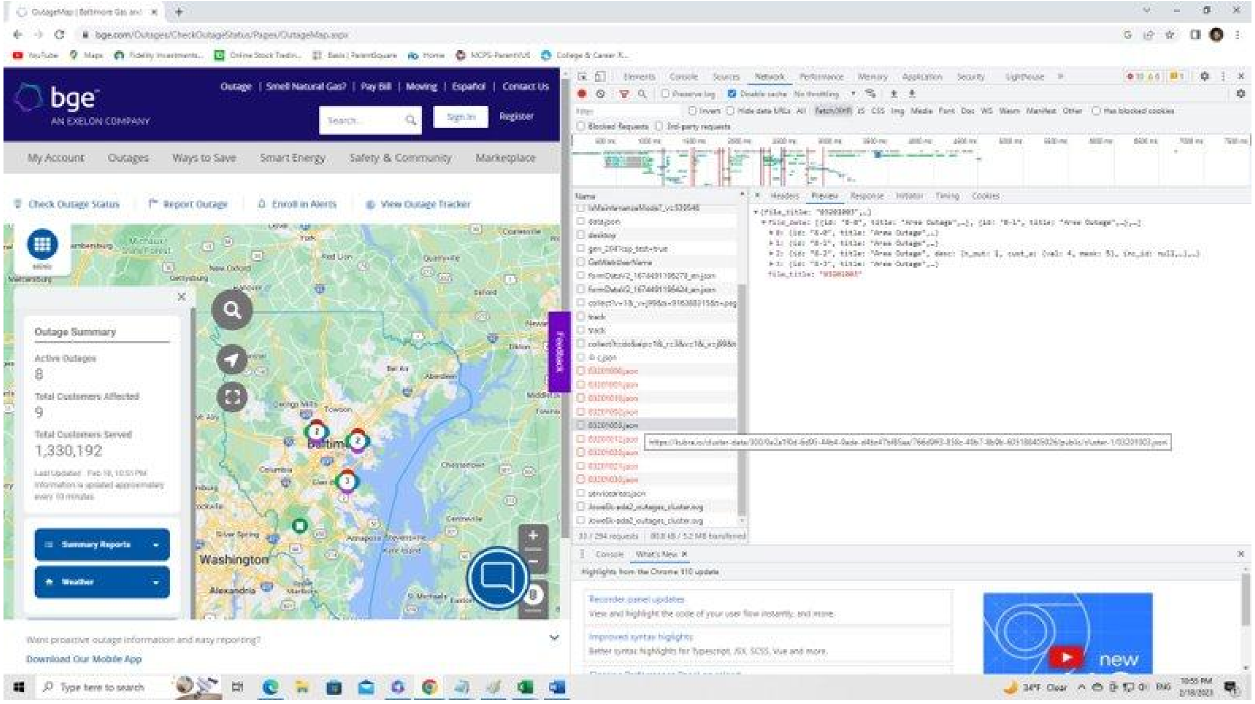


Python Requests allows scraping of surface-level HTML data, but individual outage information was stored in .json files that were initially impenetrable. The multicolored maker on the map with the number 9 actually contained 9 different outages that needed to be individually separated. These required further tools to interpret:

3. Beautiful Soup, used to extract data from HTML and XML files, as well as Selenium, used to extract data from JavaScript files.

I requested the .json script responses to get the wanted information directly. First, I obtained the specific URL address of the topmost outage marker (example: https://kubra.io/cluster-data/001/0e2a1f0d-6d95-44b4-9ade-d4bc47bf85aa/c30660df-7132-4ea2

-ae80-7f6aa8be9bd7/public/cluster-1/0320100.json). Then, I searched recursively to identify each outage contained in the parent marker and collect individual outage characteristics.


```
▼{file_title: "03201003",…}
  ▼file_data: [{id: "0-0", title: "Area Outage",…}, {id: "0-1", title: "Area Outage",…},…]
    ▶0: {id: "0-0", title: "Area Outage",…}
    ▶1: {id: "0-1", title: "Area Outage",…}
    ▶2: {id: "0-2", title: "Area Outage", desc: {n_out: 1, cust_a: {val: 4, mask: 5}, inc_id: null,…},…}
    ▶3: {id: "0-3", title: "Area Outage",…}
   file_title: "03201003"
```

Outage Summary

Active Outages
8

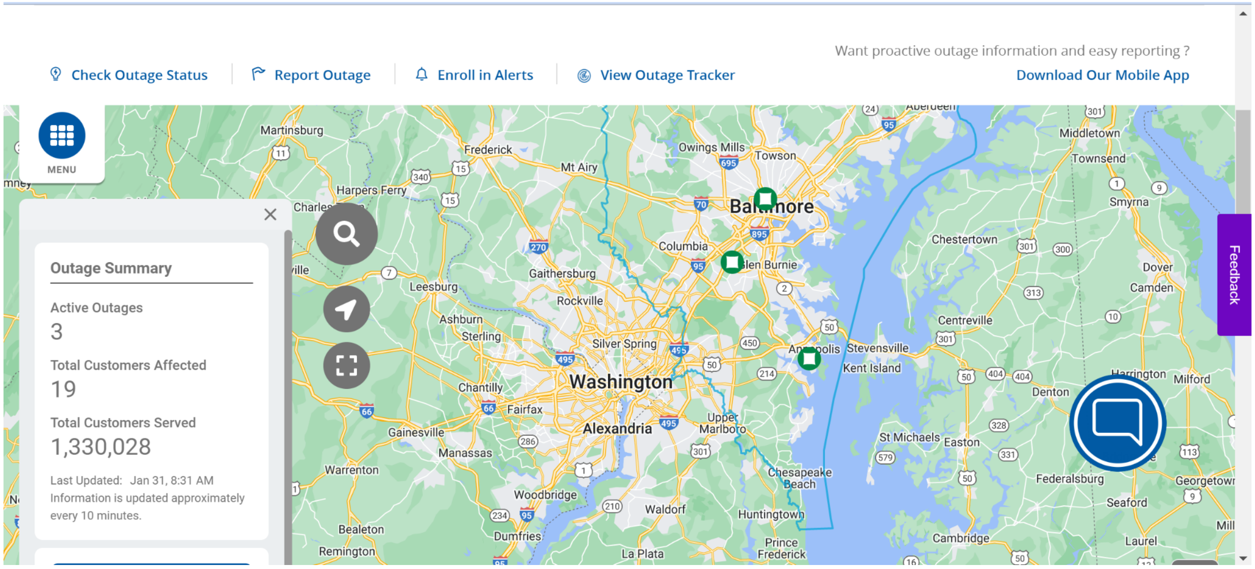
Total Customers Affected
9

Total Customers Served
1,330,192

Last Updated: Feb 18, 10:51 PM
Information is updated approximately every 10 minutes

Screenshots of the outage maps of the utilities you have web-scraped

Baltimore Gas and Electric:


Outage Summary

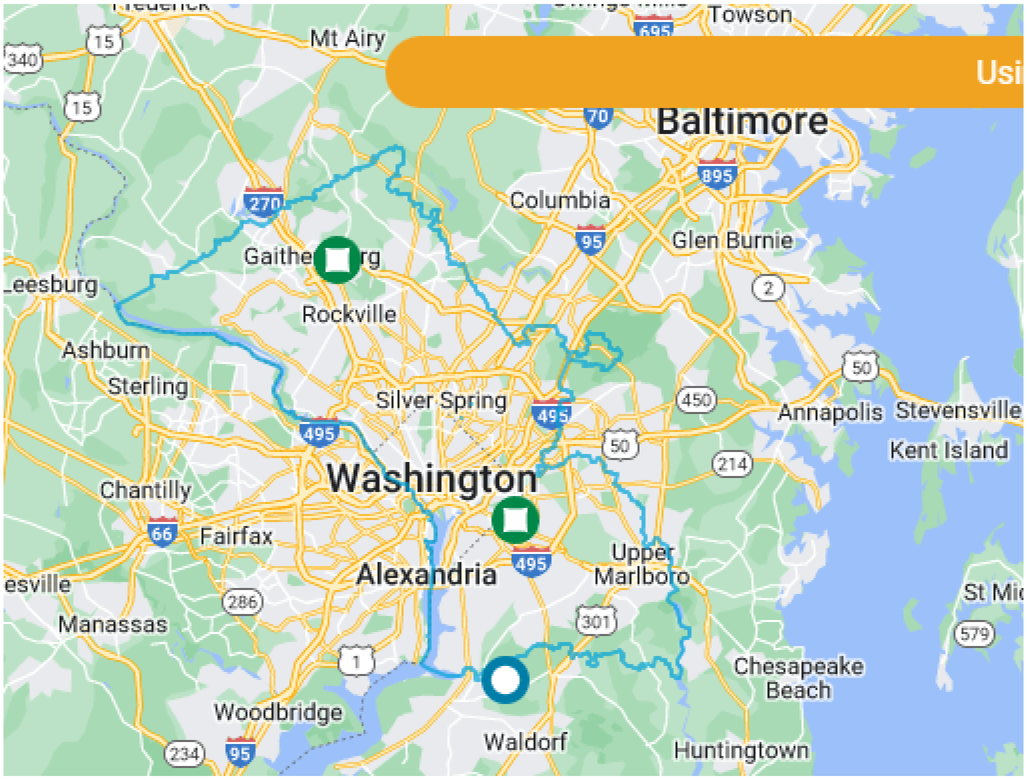
Active Outages
3

Total Customers Affected
19

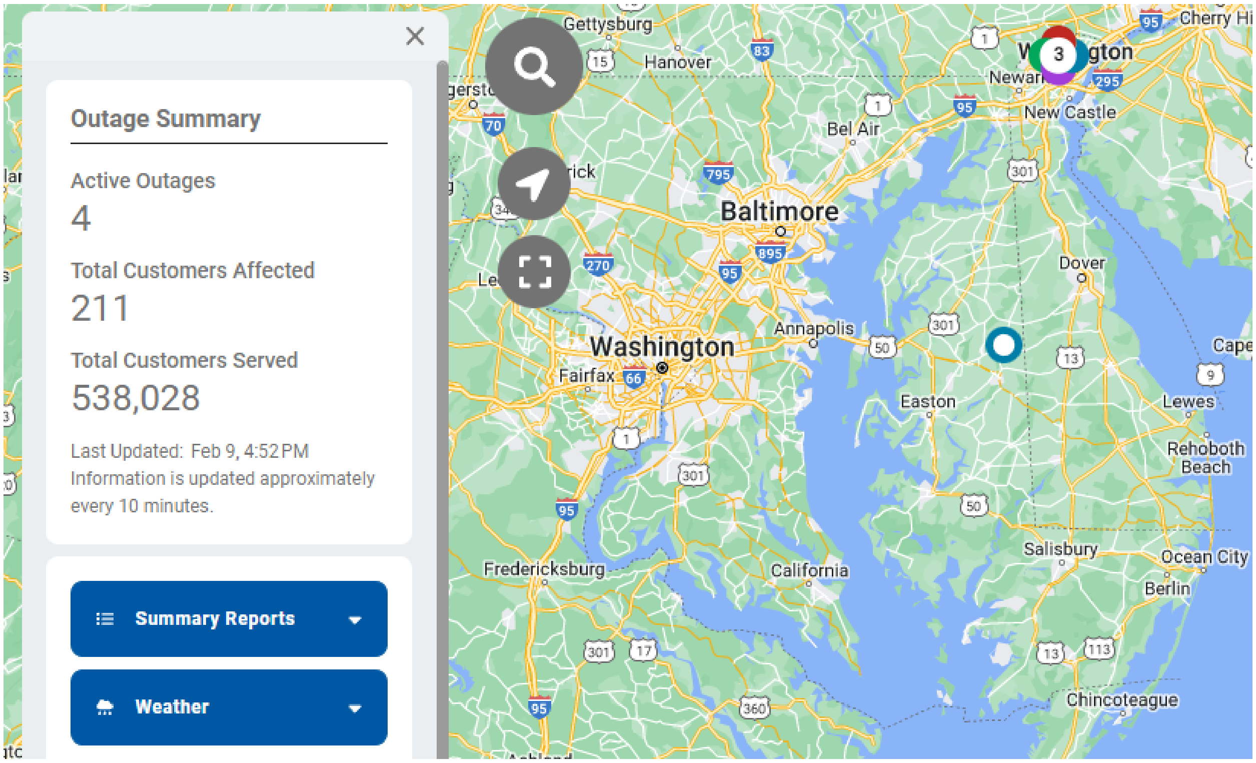
Total Customers Served
1,330,028

Last Updated: Jan 31, 8:31 AM
Information is updated approximately every 10 minutes.

PEPCO:



Delmarva Power:



Outage Summary

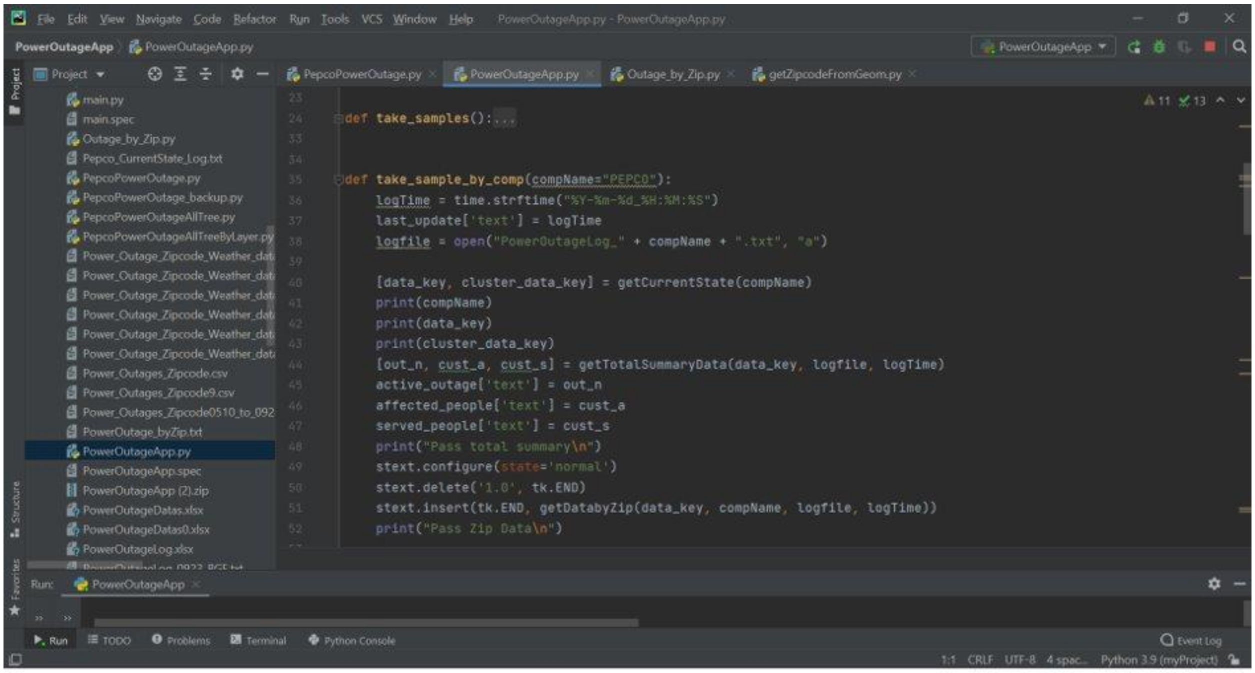
Active Outages
4

Total Customers Affected
211

Total Customers Served
538,028

Last Updated: Feb 9, 4:52 PM
Information is updated approximately every 10 minutes.

A screenshot of the python codes you have used

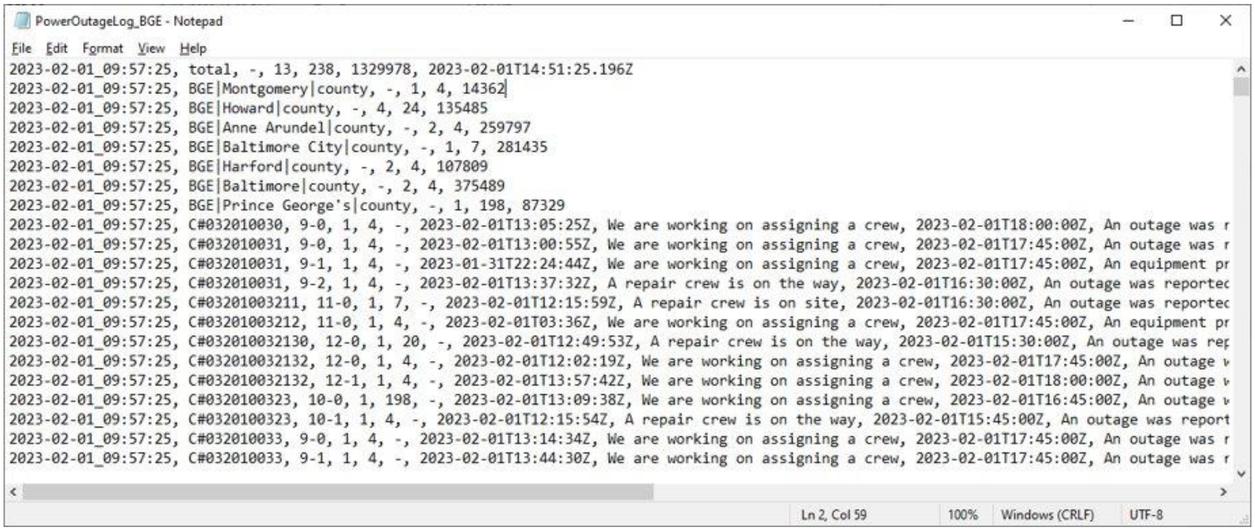

File Edit View Navigate Code Refactor Run Tools VCS Window Help
PowerOutageApp.py - PowerOutageApp.py
PowerOutageApp
PowerOutageApp.py
Project
PepcoPowerOutage.py
PowerOutageApp.py
Outage_by_Zip.py
getZipcodeFromGeom.py
main.py
main.spec
Outage_by_Zip.py
Pepco_CurrentState_Log.txt
PepcoPowerOutage.py
PepcoPowerOutage_backup.py
PepcoPowerOutageAllTree.py
PepcoPowerOutageAllTreeByLayer.py
Power_Outage_Zipcode_Weather_dat
Power_Outage_Zipcode_Weather_dat
Power_Outage_Zipcode_Weather_dat
Power_Outage_Zipcode_Weather_dat
Power_Outage_Zipcode_Weather_dat
Power_Outage_Zipcode_Weather_dat
Power_Outages_Zipcode.csv
Power_Outages_Zipcode9.csv
Power_Outages_Zipcode0510_to_092
PowerOutage_byZip.txt
PowerOutageApp.py
PowerOutageApp.spec
PowerOutageApp (2).zip
PowerOutageDatas.xlsx
PowerOutageDatas0.xlsx
PowerOutageLog.xlsx
Run: PowerOutageApp
Run
TODO
Problems
Terminal
Python Console
Event Log
1:1 CRLF UTF-8 4 spac... Python 3.9 (myProject)

```
def take_samples():...


def take_sample_by_comp(compName="PEPCO"):
    logTime = time.strftime("%Y-%m-%d_%H:%M:%S")
    last_update['text'] = logTime
    logfile = open("PowerOutageLog_" + compName + ".txt", "a")

    [data_key, cluster_data_key] = getCurrentState(compName)
    print(compName)
    print(data_key)
    print(cluster_data_key)
    [out_n, cust_a, cust_s] = getTotalSummaryData(data_key, logfile, logTime)
    active_outage['text'] = out_n
    affected_people['text'] = cust_a
    served_people['text'] = cust_s
    print("Pass total summary\n")
    stext.configure(state='normal')
    stext.delete('1.0', tk.END)
    stext.insert(tk.END, getDatabyZip(data_key, compName, logfile, logTime))
    print("Pass Zip Data\n")
```

A screenshot of the raw data directly from webscraping



```
2023-02-01_09:57:25, total, -, 13, 238, 1329978, 2023-02-01T14:51:25.196Z
2023-02-01_09:57:25, BGE|Montgomery|county, -, 1, 4, 14362
2023-02-01_09:57:25, BGE|Howard|county, -, 4, 24, 135485
2023-02-01_09:57:25, BGE|Anne Arundel|county, -, 2, 4, 259797
2023-02-01_09:57:25, BGE|Baltimore City|county, -, 1, 7, 281435
2023-02-01_09:57:25, BGE|Harford|county, -, 2, 4, 107809
2023-02-01_09:57:25, BGE|Baltimore|county, -, 2, 4, 375489
2023-02-01_09:57:25, BGE|Prince George's|county, -, 1, 198, 87329
2023-02-01_09:57:25, C#032010030, 9-0, 1, 4, -, 2023-02-01T13:05:25Z, We are working on assigning a crew, 2023-02-01T18:00:00Z, An outage was r
2023-02-01_09:57:25, C#032010031, 9-0, 1, 4, -, 2023-02-01T13:00:55Z, We are working on assigning a crew, 2023-02-01T17:45:00Z, An outage was r
2023-02-01_09:57:25, C#032010031, 9-1, 1, 4, -, 2023-01-31T22:24:44Z, We are working on assigning a crew, 2023-02-01T17:45:00Z, An equipment pr
2023-02-01_09:57:25, C#032010031, 9-2, 1, 4, -, 2023-02-01T13:37:32Z, A repair crew is on the way, 2023-02-01T16:30:00Z, An outage was reportec
2023-02-01_09:57:25, C#03201003211, 11-0, 1, 7, -, 2023-02-01T12:15:59Z, A repair crew is on site, 2023-02-01T16:30:00Z, An outage was reportec
2023-02-01_09:57:25, C#03201003212, 11-0, 1, 4, -, 2023-02-01T03:36Z, We are working on assigning a crew, 2023-02-01T17:45:00Z, An equipment pr
2023-02-01_09:57:25, C#032010032130, 12-0, 1, 20, -, 2023-02-01T12:49:53Z, A repair crew is on the way, 2023-02-01T15:30:00Z, An outage was rep
2023-02-01_09:57:25, C#032010032132, 12-0, 1, 4, -, 2023-02-01T12:02:19Z, We are working on assigning a crew, 2023-02-01T17:45:00Z, An outage w
2023-02-01_09:57:25, C#032010032132, 12-1, 1, 4, -, 2023-02-01T13:57:42Z, We are working on assigning a crew, 2023-02-01T18:00:00Z, An outage w
2023-02-01_09:57:25, C#0320100323, 10-0, 1, 198, -, 2023-02-01T13:09:38Z, We are working on assigning a crew, 2023-02-01T16:45:00Z, An outage w
2023-02-01_09:57:25, C#0320100323, 10-1, 1, 4, -, 2023-02-01T12:15:54Z, A repair crew is on the way, 2023-02-01T15:45:00Z, An outage was report
2023-02-01_09:57:25, C#032010033, 9-0, 1, 4, -, 2023-02-01T13:14:34Z, We are working on assigning a crew, 2023-02-01T17:45:00Z, An outage was r
2023-02-01_09:57:25, C#032010033, 9-1, 1, 4, -, 2023-02-01T13:44:30Z, We are working on assigning a crew, 2023-02-01T17:45:00Z, An outage was r
```

The scope of web-scraped data you have collected so far

Our data covers 3 utilities that service around 2.7 million customers in total. The location, duration, stated cause, and estimated restoration time of outages that were reported on the outage maps of the 3 utilities in the following time periods were captured:

2022/06/28 to 2022/12/08

2023/02/01 to 2023/02/21

Data was collected every 10 minutes. Events when my computer lost internet caused us to lose outage data during those time periods.

6). The types of variables and outage information included in the web-scraped data

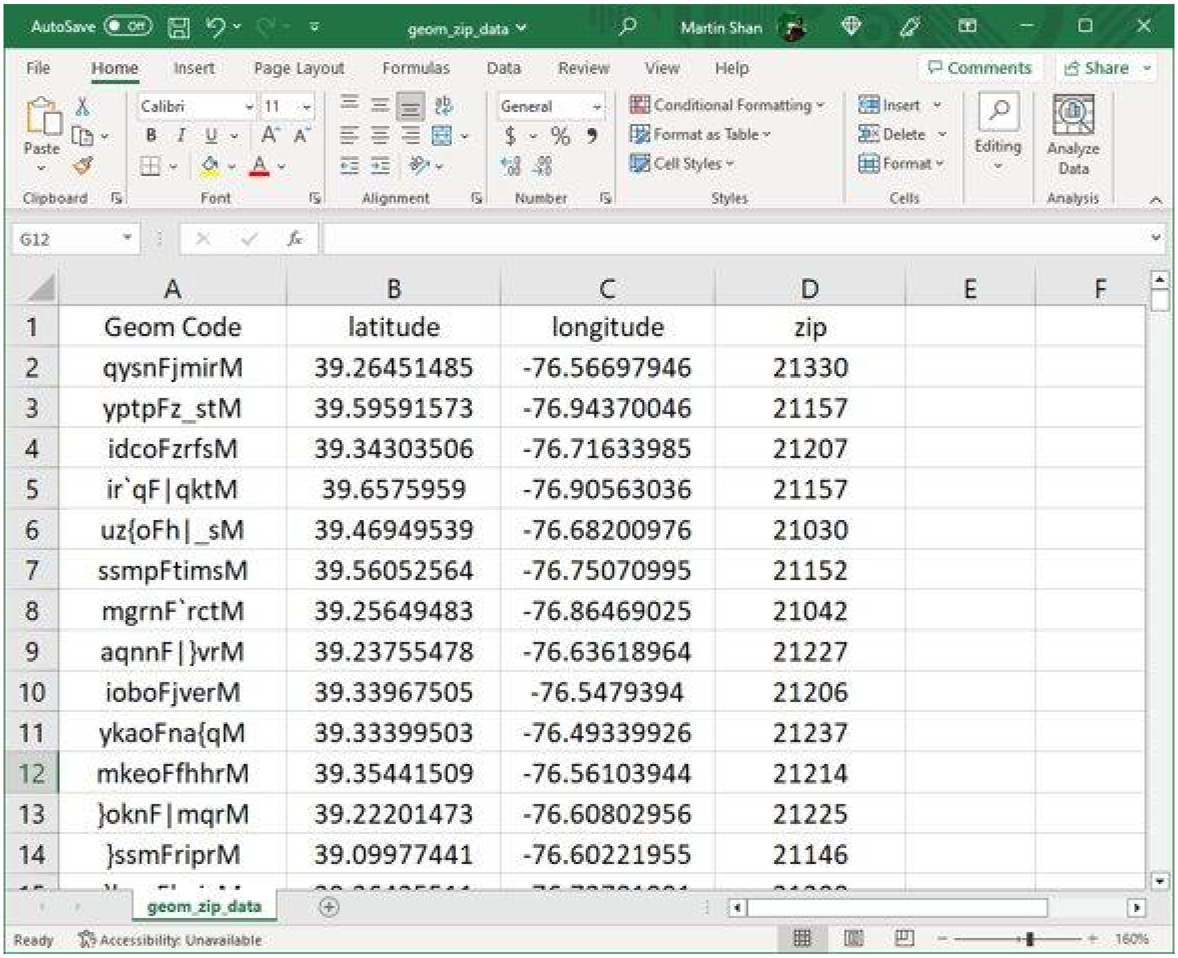
Example:

- Data Log Time: 2023-02-01_09:57:25,
- LogFile#: C#032010033,
- Outage ID#: 9-1,
- Outage No.: 1,
- Affected: 4,
- Served: -,
- Report Time: 2023-02-01T13:44:30Z,
- Action: We are working on assigning a crew,
- Expected Fix Time: 2023-02-01T17:45:00Z,
- Reason: An outage was reported in your area.
- GeomCode: entlFrxzqM, (later converted to latitude and longitude)

Deciphering GeomCode:
Initially, all outage data was encrypted in 10-character strings similar to “entlFrxzqM”. To decode these strings, I downloaded hundreds of GeomCode examples from outage maps representing multiple companies. My examples covered as far south as Florida and Texas, as far north as New Hampshire, as far east as Boston, and as far west as Chicago.

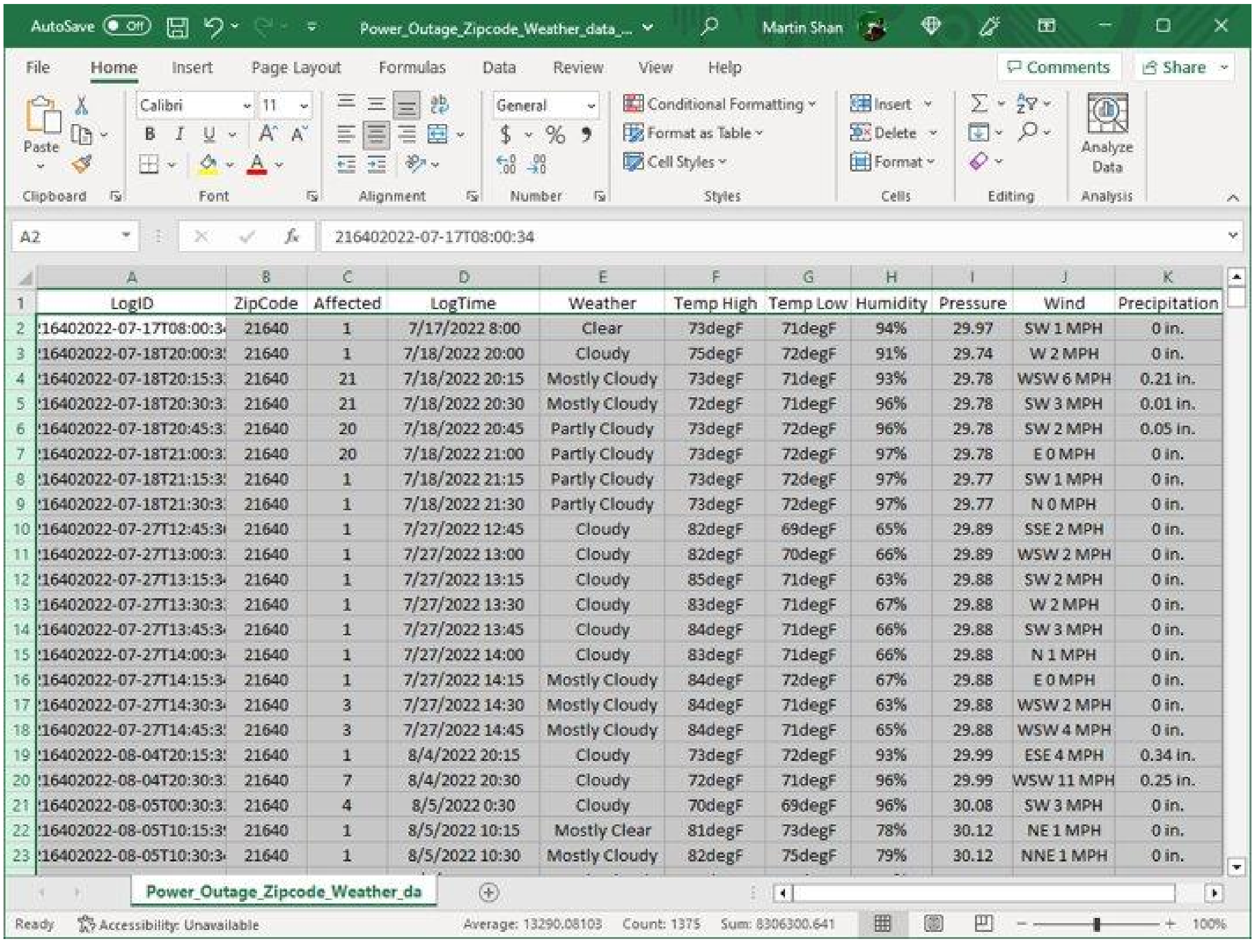
I recorded the location of each datapoint using Google Map to correlate each character of GeomCode with Latitude and Longitude. Using these figures, I reverse engineered the algorithm that turned an input of the base 32 interpretation of each character into a latitude and longitude.

I then used GeoPy, a Python library, to convert the latitude and longitude data into zip codes. A chart listing various GeomCodes, their lat/long interpretations, and their corresponding zip codes is shown below.



	A	B	C	D	E	F
1	Geom Code	latitude	longitude	zip		
2	qysnFjmirM	39.26451485	-76.56697946	21330		
3	yptpFz_stM	39.59591573	-76.94370046	21157		
4	idcoFzrfsM	39.34303506	-76.71633985	21207		
5	ir`qF|qktM	39.6575959	-76.90563036	21157		
6	uz{oFh|_sM	39.46949539	-76.68200976	21030		
7	ssmpFtimsM	39.56052564	-76.75070995	21152		
8	mgrnF`rctM	39.25649483	-76.86469025	21042		
9	aqnnF|}vrM	39.23755478	-76.63618964	21227		
10	ioboFjverM	39.33967505	-76.5479394	21206		
11	ykaoFna{qM	39.33399503	-76.49339926	21237		
12	mkeoFfhhrM	39.35441509	-76.56103944	21214		
13	}oknF|mqrM	39.22201473	-76.60802956	21225		
14	}ssmFriprM	39.09977441	-76.60221955	21146		

Using this location data, I matched each outage to corresponding weather data using data webscraped from weatherforyou.com. A screenshot of the variables is shown below:



A2 | 216402022-07-17T08:00:34

	A	B	C	D	E	F	G	H	I	J	K
1	LogID	ZipCode	Affected	LogTime	Weather	Temp High	Temp Low	Humidity	Pressure	Wind	Precipitation
2	16402022-07-17T08:00:3	21640	1	7/17/2022 8:00	Clear	73degF	71degF	94%	29.97	SW 1 MPH	0 in.
3	16402022-07-18T20:00:3	21640	1	7/18/2022 20:00	Cloudy	75degF	72degF	91%	29.74	W 2 MPH	0 in.
4	16402022-07-18T20:15:3	21640	21	7/18/2022 20:15	Mostly Cloudy	73degF	71degF	93%	29.78	WSW 6 MPH	0.21 in.
5	16402022-07-18T20:30:3	21640	21	7/18/2022 20:30	Mostly Cloudy	72degF	71degF	96%	29.78	SW 3 MPH	0.01 in.
6	16402022-07-18T20:45:3	21640	20	7/18/2022 20:45	Partly Cloudy	73degF	72degF	96%	29.78	SW 2 MPH	0.05 in.
7	16402022-07-18T21:00:3	21640	20	7/18/2022 21:00	Partly Cloudy	73degF	72degF	97%	29.78	E 0 MPH	0 in.
8	16402022-07-18T21:15:3	21640	1	7/18/2022 21:15	Partly Cloudy	73degF	72degF	97%	29.77	SW 1 MPH	0 in.
9	16402022-07-18T21:30:3	21640	1	7/18/2022 21:30	Partly Cloudy	73degF	72degF	97%	29.77	N 0 MPH	0 in.
10	16402022-07-27T12:45:3	21640	1	7/27/2022 12:45	Cloudy	82degF	69degF	65%	29.89	SSE 2 MPH	0 in.
11	16402022-07-27T13:00:3	21640	1	7/27/2022 13:00	Cloudy	82degF	70degF	66%	29.89	WSW 2 MPH	0 in.
12	16402022-07-27T13:15:3	21640	1	7/27/2022 13:15	Cloudy	85degF	71degF	63%	29.88	SW 2 MPH	0 in.
13	16402022-07-27T13:30:3	21640	1	7/27/2022 13:30	Cloudy	83degF	71degF	67%	29.88	W 2 MPH	0 in.
14	16402022-07-27T13:45:3	21640	1	7/27/2022 13:45	Cloudy	84degF	71degF	66%	29.88	SW 3 MPH	0 in.
15	16402022-07-27T14:00:3	21640	1	7/27/2022 14:00	Cloudy	83degF	71degF	66%	29.88	N 1 MPH	0 in.
16	16402022-07-27T14:15:3	21640	1	7/27/2022 14:15	Mostly Cloudy	84degF	72degF	67%	29.88	E 0 MPH	0 in.
17	16402022-07-27T14:30:3	21640	3	7/27/2022 14:30	Mostly Cloudy	84degF	71degF	63%	29.88	WSW 2 MPH	0 in.
18	16402022-07-27T14:45:3	21640	3	7/27/2022 14:45	Mostly Cloudy	84degF	71degF	65%	29.88	WSW 4 MPH	0 in.
19	16402022-08-04T20:15:3	21640	1	8/4/2022 20:15	Cloudy	73degF	72degF	93%	29.99	ESE 4 MPH	0.34 in.
20	16402022-08-04T20:30:3	21640	7	8/4/2022 20:30	Cloudy	72degF	71degF	96%	29.99	WSW 11 MPH	0.25 in.
21	16402022-08-05T00:30:3	21640	4	8/5/2022 0:30	Cloudy	70degF	69degF	96%	30.08	SW 3 MPH	0 in.
22	16402022-08-05T10:15:3	21640	1	8/5/2022 10:15	Mostly Clear	81degF	73degF	78%	30.12	NE 1 MPH	0 in.
23	16402022-08-05T10:30:3	21640	1	8/5/2022 10:30	Mostly Cloudy	82degF	75degF	79%	30.12	NNE 1 MPH	0 in.

7). How promising are we able to web-scrape similar data from other utilities located in other states?

Exelon Corp. is the parent company of the 3 utilities that I webscraped (BGE, PEPCO, Delmarva). All 3 utilities used identical website infrastructure, which allowed for a single scraping program to collect data from all 3 websites.

There are about 55 similar parent companies in the US. Each company operates between 1 and 11 utilities, each with their separate websites, for a total of 128 separate websites.

Many utilities on the east coast use the same base data provider, Kubra, as Exelon, and can be scraped using similar methods. However, utilities in other parts of the country, such as Pacific Gas and Electric in California, have websites that are constructed very differently. Some examples:

Company	Area	Database
PGE.com	Central CA	Esri's GIS
Somersetrec	Western PA	Esri's GIS
TNMP	Oklahoma City	Microsoft Bing Maps
Pacific Power	California, Washington, Oregon	Google Maps API

In these cases, it may not be possible to scrape outages from the utility websites, and the amount of time to develop a program suited to scraping a specific database may vary largely, from days to months.

8). What would be the feasible number of utilities we will be able to webscrape within 3 years?

10 companies is likely possible - total number of utilities depends on the website structure of each company and the amount of utilities each company has under management. Going beyond this number depends heavily on the privacy of each database structure. The US is split into 22 regions by eGRID. Since companies have multiple utilities under management on average, it may be possible to cover every region using these 10 companies.

9). The types of computing and data storage resources needed for scraping the number of utilities mentioned in 8).

Two computers with GPUs should be sufficient for webscraping, provided that databases besides Kudra require similar amounts of computation time to scrape.

Collected data shows about 6 Gb data/month on average and a peak of 33 Gb/month for BGE, which has 1.3 million customers. We might expect 150 Gb/year/1 million customers. Aiming for 100 million, we would require around 45 Tb for 3 years of data.