SI 206 Final Project Report:

Analyzing the Impact of Temperature on COVID-19 and Flu Trends in Michigan and Nationwide

[GitHub Final Project Repository]
Fall 2024 – Data-Oriented Programming
Tharron Combs and Ashley Xu

Original Research Question:
Research question: How does temperature affect flu and COVID-19 cases in Michigan and nationwide?

1. Goals (20 Points)

Initial Goals of Project, Based on the Research Question:

1. **Overview of project goals:**
   - Retrieve data from three APIs: Delphi Epidata, **Covid Act Now API**, and Meteostat JSON API.
   - Find data from the time frame of March 2020 (the onset of COVID-19 in the US) to March 2023.
   - Analyze trends in flu (`num_ili` from Delphi API) and COVID-19 case counts alongside temperature data (TAVG fom Meteostat JSON API).
   - Create visualizations to demonstrate seasonal trends and relationships between temperature and illness.
2. **Data Collection:**
   - Retrieve and store data on weekly flu cases (measured by `num_ili`) in Michigan and nationally.
   - Retrieve and store data on weekly confirmed COVID-19 cases in Michigan and nationally.
   - Retrieve and store daily temperature data for Michigan, including average temperature (TAVG), maximum temperature (TMAX), and minimum temperature (TMIN).
3. **Data Integration and Analysis:**
   - Correlate flu cases (weekly `num_ili`) in Michigan with temperature trends (`TAVG`) over time.
   - Correlate COVID-19 cases with temperature data in Michigan over time.
   - Investigate relationships between national trends in flu cases and temperature data.
4. **Seasonal Trends:**
   - Identify seasonal trends in flu (`num_ili`) and COVID-19 case counts in Michigan.
   - Use visualizations to show whether temperature fluctuations have a correlation with flu and COVID-19 trends in reported cases.
5. **Visualization and Reporting:**
   - Create clear visualizations showing relationships between temperature and flu/COVID-19 case counts.
   - Highlight seasonal patterns for flu and COVID-19 case trends in Michigan and nationally.

Achieved Goals

1. **Overview of achieved goals**
    - Successfully retrieved, processed, and stored weekly flu, COVID-19, and temperature data in an SQLite database.
    - Analyzed weekly flu trends and correlated them with temperature in Michigan.
    - Generated visualizations for flu trends and temperature relationships, achieving clear insights into the impact of temperature on public health (measured by flu and COVID-19 case trends).
2. **Data Retrieval and Storage:**
    - Retrieved weekly flu case data (`num_ili` and `wili`) for Michigan and nationally from the Delphi Epidata API.
    - Retrieved daily COVID-19 case data for Michigan and nationally from the COVID-19 Statistics API.
    - Retrieved daily temperature data (TAVG, TMAX, TMIN) for Michigan from the Climate Data Online (CDO) API.
    - Stored all data in an SQLite database with shared keys for integration (`date` and `region`).
3. **Analysis and Calculations:**
    - Calculated weekly averages for flu (`num_ili`) and COVID-19 cases
    - Calculated average weekly temperature by averaging daily temperature
    - Identified seasonal trends in flu cases and COVID-19 case counts in Michigan and nationally.
4. **Visualizations:**
    - Used **line charts** to represent data from March 2020 to March 2022:
        - Used shaded blue and red backgrounds to identify the summer and winter months
            - Created one line chart for weekly new flu cases trends in Michigan and nationally.
            - Created one line chart for weekly new COVID-19 cases trends in Michigan and nationally.
    - Created one line chart showing the average national weekly temperature
    - Created one line chart showing the average Michigan weekly temperature
5. **Reporting and Statistical Interpretation:**
    - Visually presented the results of the data compiled from APIs, including displaying the number of new flu and COVID-19 cases and pointing out trends during winter and summer months.

Division of Workload

- Ashley Xu: Delphi Epidata API implementation and 2 corresponding visualizations representing flu trends.
- Tharron Combs: COVID-19 API and 2 corresponding visualizations representing COVID-19 trends nationally .
- Shared tasks: Database setup, Climate data API, and final report

2. Problems Faced (10 Points)

Data Retrieval Challenges:

- Our initial chosen APIs for COVID-19 cases and weather data posed difficulties in retrieval and analysis:
    - The COVID-19 API we originally chose was poorly documented, making data extraction and processing cumbersome.
    - The Climate Data Online (CDO) API we originally chose from the National Centers for Environmental Information (NCEI) presented challenges due to its aggressive rate limits, which significantly slowed the retrieval of Michigan temperature data.
        - This API also had an overly complicated structure and did not support geo-location-based data. It provided regional averages instead of state-specific data, which conflicted with our project's focus on Michigan.
- Determining a national average for weather data using the weather API was challenging. We addressed this by selecting major cities across NCEI-defined climatic regions and calculating an average temperature. These cities were chosen based on their large populations and regional representation to reflect national trends accurately.

Data Processing Challenges:

- Aligning dates across the datasets retrieved from different APIs (flu cases, COVID-19 cases, and weather) was a significant challenge. Each API used different date formats and time zones. To resolve this, we standardized all data to Coordinated Universal Time (UTC) format to ensure data uniformity in the database for accurate analysis.
    - For example, the Delphi Epidata API gave us data on the number of new flu cases in each epiweek (or epidemiological week which is typically measured/numbered by the first full week of the year)
- COVID-19 data was provided as cumulative totals, requiring preprocessing to convert the data into weekly counts for trend analysis.
- Aligning flu and temperature data required careful filtering to account for missing or incomplete temperature records, ensuring data accuracy for the analysis.

Changes to Region Data: From Strings to Integers

As part of the data cleaning and enhancement process, we modified how regional data was stored in the database. Initially, the regions (e.g., "MI" for Michigan and "National") were represented as strings in the database. However, Replaced string identifiers (e.g., 'MI' for Michigan) with integer keys (e.g., '1' for Michigan) to improve storage efficiency and query performance. The new system uses integers like "1" for Michigan and "2" for National data. This change ensures consistency and enhances the efficiency of database operations, especially when joining tables or performing aggregation tasks.

Visualization Challenges for Line Charts:

We encountered difficulties with deciding on the best methods to visually represent data from different APIs (e.g., flu trends, COVID-19 trends, and temperature correlations)

To best represent our data in a digestible, intuitive format, we addressed these challenges by:

- Adjusting x-axis tick labels to focus on key months (March, June, September, December)
- Using commas for thousands (of flu or COVID-19 cases) on the y-axis to improve clarity and readability.
- Using dots to represent the number of cases or distinct temperature from each week of the year
- Choosing appropriate colors for line plots (e.g., flu and COVID-19 trends of new cases)

- We used shaded background bars to highlight the winter and summer months of the year, using red to represent the summer (warmer) months and blue to represent winter (colder) months
- We used these colors to illustrate an intuitive and readable understanding of our data

3. Limitations of Our Research and Data

While the project provides valuable insights into flu and COVID-19 trends, there are some limitations to consider:

Impact of COVID-19 on Flu Data:
The COVID-19 pandemic may have impacted the number of reported influenza-like illnesses (ILI). Many health systems and public health organizations have noted that flu cases were lower during the COVID-19 pandemic, possibly due to measures like social distancing, mask-wearing, and increased hygiene practices. These factors could skew the comparison between flu and COVID-19 trends.

Additionally, the categorization of ILI data could overlap with COVID-19 symptoms, making it difficult to distinguish between the two diseases, especially in regions where COVID-19 testing was limited or delayed.

Incomplete or Missing Data:
While the data sources provide comprehensive datasets, missing or incomplete data can impact the quality of the analysis. For example, some regions may have incomplete temperature data, or COVID-19 case reporting may be inconsistent across time and geography.

API Limitations:
The data collected relies on APIs, which are sometimes subject to rate limits or data availability. If API access is interrupted or data becomes unavailable, the completeness of the dataset may be compromised.

Geographic and Temporal Scope:
The analysis focuses on Michigan and national trends in the United States. This geographic limitation means the findings may not be applicable globally or to other regions with different public health measures or climate patterns. Additionally, the dataset spans only certain time frames (e.g., March 2020 to March 2022), and trends could change over longer periods.

Potential Bias in Flu Reporting:
Flu case reporting may vary from year to year, and certain factors such as changes in healthcare access, public awareness, and diagnostic practices could lead to underreporting or overreporting of flu cases, particularly in the pre-COVID era.

4. Calculation File (10 Points)

Link:
https://github.com/duckgandalfsaxophone/SI206FinalProject/blob/4a7ea00fc7c6e71e46d5b25f75e0c47e59ab4def/data_collection.py

The file contains all calculations, including weekly flu trends, COVID-19 case averages, and averages calculated for temperature data

5. Overview of Databases Created

In this project, we created a series of SQLite databases to store the processed data retrieved from various APIs. These databases are integral to organizing and analyzing data related to flu cases, COVID-19 cases, and temperature trends. Below is an overview of the main tables (with screenshots included below) created in the project, along with the shared keys used across the databases:

Flu Data Table (table: flu_data_march_2020_to_2023)
Description: This table stores weekly flu case data, including week_id, num_ili (flu cases), and region_key. The week_id serves as a shared key to link flu data with other sources, enabling weekly aggregation and analysis.

Weekly COVID-19 Data Table (tables: weekly_michigan_covid_data and weekly_national_covid_data)
Description: Similar to the flu data table, this table contains weekly COVID-19 case counts. The week_id is used to aggregate cases on a weekly basis, ensuring consistency in analysis. Separate tables are maintained for Michigan and national data.

Temperature Data Table (tables: michigan_weather_data and national_weather_data)
Description: These tables store weekly temperature data (including TAVG, TMAX, and TMIN temperatures) for Michigan and national locations. The week_id serves as the common key to align temperature data with flu and COVID-19 case counts, enabling cross-comparison across datasets.

Daily National and Michigan COVID-19 Data Table (tables: daily_national_covid_data and daily_michigan_covid_data)
Description: These tables contain raw, daily confirmed cases of COVID-19, with each entry representing the cumulative count of cases up to that day. The data does not initially include a week_id field, which would typically facilitate weekly grouping and aggregation. Therefore, we generated the week_id based on the date and aggregated the cases into weekly totals.

Run Count Table (table: run_counts)
Description: This table tracks how many times each data collection task has been run. It is crucial for managing API calls and ensuring that data is collected in stages, especially by limiting the retrieval to 25 rows per run and ensuring the completion of data collection on the fifth run.

Shared Keys and IDs

- **week_id**: This key is used across all tables to uniquely identify each week in the dataset. It ensures that flu cases, COVID-19 cases, and temperature data are linked by the same time period (weekly), making it easier to correlate trends across datasets.
- **location_key / region_key:** These keys were used to differentiate between Michigan and national data. To optimize database efficiency and ensure consistency, the region identifiers (e.g., "MI" for Michigan and "National" for the U.S.) were replaced with integer keys (e.g., "1" for Michigan and "2" for National data). This also reduces redundancy and improves query performance.
- **run_count**: This key tracks how many times the data collection tasks have been performed, which is vital for controlling the flow of data retrieval. It ensures that only partial data is fetched during the first few runs, with all data being collected on the fifth run to avoid unnecessary duplication.

These shared keys help organize and maintain consistency in the database, allowing for efficient integration of data from different sources. The relationships between these tables are crucial for analyzing the impact of temperature on flu and COVID-19 trends, and they ensure the integrity and reliability of the data used for analysis.

(Screenshots of the first few rows of each database table are included below.)

Table: daily_national_covid_data

	date	cases
	Filter	Filter
1	2020-03-09	NULL
2	2020-03-10	NULL
3	2020-03-11	1263
4	2020-03-12	1668
5	2020-03-13	2224
6	2020-03-14	2897
7	2020-03-15	3596
8	2020-03-16	4502
9	2020-03-17	5901
10	2020-03-18	8339
11	2020-03-19	12378
12	2020-03-20	17992
13	2020-03-21	24507
14	2020-03-22	33029
15	2020-03-23	43459
16	2020-03-24	53889
17	2020-03-25	68523
18	2020-03-26	85504
19	2020-03-27	102828
20	2020-03-28	123890
21	2020-03-29	142405
22	2020-03-30	163865
23	2020-03-31	188292
24	2020-04-01	215214
25	2020-04-02	244919
26	2020-04-03	277234
27	2020-04-04	312260
28	2020-04-05	337834
29	2020-04-06	368747
30	2020-04-07	399075

1 - 30 of 1,511

Table: daily_michigan_covid_data

	date	cases
	Filter	Filter
1	2020-03-04	NULL
2	2020-03-05	NULL
3	2020-03-06	NULL
4	2020-03-07	NULL
5	2020-03-08	NULL
6	2020-03-09	NULL
7	2020-03-10	2
8	2020-03-11	2
9	2020-03-12	12
10	2020-03-13	25
11	2020-03-14	33
12	2020-03-15	53
13	2020-03-16	54
14	2020-03-17	65
15	2020-03-18	80
16	2020-03-19	334
17	2020-03-20	548
18	2020-03-21	787
19	2020-03-22	1033
20	2020-03-23	1324
21	2020-03-24	1791
22	2020-03-25	2293
23	2020-03-26	2877
24	2020-03-27	3655
25	2020-03-28	4634
26	2020-03-29	5486
27	2020-03-30	6508
28	2020-03-31	7629
29	2020-04-01	9292
30	2020-04-02	10791

1 - 30 of 1,548

Table: flu_data_march_2020_to_2023

	region_key	date	week_id	num_ili
	Filter	Filter	Filter	Filter
1	1	2020-03-01	202009	736
2	2	2020-03-01	202009	78779
3	1	2020-03-08	202010	531
4	2	2020-03-08	202010	89607
5	1	2020-03-15	202011	260
6	2	2020-03-15	202011	78584
7	1	2020-03-22	202012	111
8	2	2020-03-22	202012	53142
9	1	2020-03-29	202013	101
10	2	2020-03-29	202013	36251
11	1	2020-04-05	202014	101
12	2	2020-04-05	202014	24974
13	1	2020-04-12	202015	69
14	2	2020-04-12	202015	18274
15	1	2020-04-19	202016	88
16	2	2020-04-19	202016	15164
17	1	2020-04-26	202017	60
18	2	2020-04-26	202017	12821
19	1	2020-05-03	202018	37
20	2	2020-05-03	202018	11058
21	1	2020-05-10	202019	46
22	2	2020-05-10	202019	10480
23	1	2020-05-17	202020	17
24	2	2020-05-17	202020	11416
25	1	2020-05-24	202021	72
26	2	2020-05-24	202021	9605
27	1	2020-05-31	202022	59
28	2	2020-05-31	202022	9318
29	1	2020-06-07	202023	68
30	2	2020-06-07	202023	9572

1 - 30 of 316

Table: michigan_weather_data

	week_id	tavg_f	tmin_f	tmax_f
	Filter	Filter	Filter	Filter
1	202008	29.66	17.42	41.9
2	202009	38.2228571428571	29.3257142857143	47.0428571428571
3	202010	40.4085714285714	32.4628571428571	48.2257142857143
4	202011	38.1714285714286	27.2942857142857	48.9714285714286
5	202012	43.52	36.2171428571429	50.7457142857143
6	202013	43.6485714285714	34.8542857142857	52.3142857142857
7	202014	47.8657142857143	37.3742857142857	58.28
8	202015	37.8114285714286	28.0657142857143	47.4285714285714
9	202016	41.6942857142857	32.3857142857143	50.9257142857143
10	202017	55.8114285714286	44.8828571428571	66.6885714285714
11	202018	44.6	33.9542857142857	55.1685714285714
12	202019	51.3885714285714	41.8742857142857	60.8
13	202020	61.4942857142857	53.8314285714286	69.0542857142857
14	202021	67.9742857142857	57.9971428571429	77.9
15	202022	69.6971428571429	57.4828571428571	81.86
16	202023	67.0742857142857	55.2714285714286	78.8257142857143
17	202024	71.7542857142857	58.7428571428571	84.6628571428571
18	202025	72.1657142857143	62.2142857142857	82.04
19	202026	76.6914285714286	65.0171428571429	88.2371428571429
20	202027	79.2114285714286	68.6428571428571	89.7028571428571
21	202028	74.6085714285714	65.5571428571428	83.5314285714286
22	202029	74.9685714285714	65.9428571428572	83.9171428571428
23	202030	73.58	65.1457142857143	81.9371428571429
24	202031	69.7228571428571	59.54	79.16
25	202032	73.9914285714286	63.6285714285714	84.2
26	202033	70.52	59.18	81.8085714285714
27	202034	73.6314285714286	64.76	82.4514285714286
28	202035	67.4085714285714	55.8885714285714	78.0542857142857
29	202036	64.7857142857143	58.3057142857143	71.1628571428571
30	202037	56.0428571428571	45.6285714285714	66.38

1 - 30 of 161

Table: national_weather_data

	time	tavg	tmin	tmax
	Filter	Filter	Filter	Filter
1	2020-03-01	-1.3	-8.1	5.5
2	2020-03-02	6.8	2.5	11.1
3	2020-03-03	4.2	-1.4	9.8
4	2020-03-04	4.0	-0.2	8.2
5	2020-03-05	1.6	-4.5	7.6
6	2020-03-06	0.6	-1.7	2.8
7	2020-03-07	-0.2	-5.3	4.9
8	2020-03-08	7.2	0.2	14.1
9	2020-03-09	11.4	6.2	16.5
10	2020-03-10	4.8	-2.2	11.7
11	2020-03-11	2.2	-2.4	6.8
12	2020-03-12	6.8	2.9	10.6
13	2020-03-13	5.1	0.6	9.5
14	2020-03-14	0.9	-0.3	2.1
15	2020-03-15	1.5	-3.0	5.9
16	2020-03-16	2.0	-4.2	8.2
17	2020-03-17	4.6	-1.9	11.0
18	2020-03-18	3.1	-2.3	8.4
19	2020-03-19	9.4	3.8	14.9
20	2020-03-20	8.2	-2.2	18.5
21	2020-03-21	-1.9	-5.1	1.3
22	2020-03-22	-1.4	-6.4	3.7
23	2020-03-23	2.8	0.2	5.4
24	2020-03-24	3.7	0.7	6.6
25	2020-03-25	5.6	-0.8	12.0
26	2020-03-26	8.8	3.2	14.3
27	2020-03-27	4.2	1.4	6.9
28	2020-03-28	7.0	4.2	9.8
29	2020-03-29	12.7	7.5	17.9
30	2020-03-30	5.9	3.7	8.1

1 - 30 of 1,096

Table: run_counts

	table_name	run_count
	Filter	Filter
1	national_weather_data	5
2	weekly_michigan_covid_data	5
3	weekly_national_covid_data	5
4	flu_data_march_2020_to_2023	5

Table: weekly_michigan_covid_data

	week_id	weekly_cases
	Filter	Filter
1	202010	51
2	202011	980
3	202012	4453
4	202013	10147
5	202014	8863
6	202015	6853
7	202016	6654
8	202017	5800
9	202018	3313
10	202019	3938
11	202020	3562
12	202021	2739
13	202022	7260
14	202023	1631
15	202024	1627
16	202025	2169
17	202026	2985
18	202027	3909
19	202028	4999
20	202029	4815
21	202030	5107
22	202031	4978
23	202032	5488
24	202033	4552
25	202034	5723
26	202035	4712
27	202036	5765
28	202037	5073
29	202038	6261
30	202039	6878

1 - 30 of 159

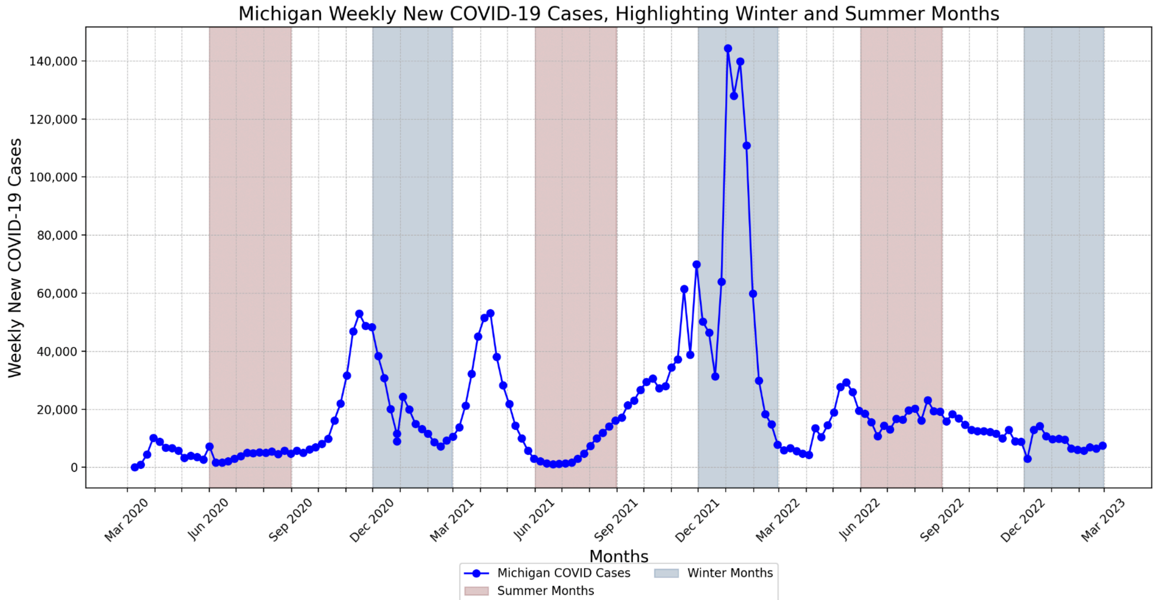
Table: weekly_national_covid_data

	week_id	weekly_cases
	Filter	Filter
1	202010	2333
2	202011	29433
3	202012	109376
4	202013	195429
5	202014	219648
6	202015	199253
7	202016	213205
8	202017	194218
9	202018	171481
10	202019	156935
11	202020	157693
12	202021	147821
13	202022	154127
14	202023	153015
15	202024	187916
16	202025	270992
17	202026	345275
18	202027	407379
19	202028	466753
20	202029	458902
21	202030	434271
22	202031	376968
23	202032	354847
24	202033	292008
25	202034	294366
26	202035	278380
27	202036	243261
28	202037	281373
29	202038	289410
30	202039	301974

1 - 30 of 159

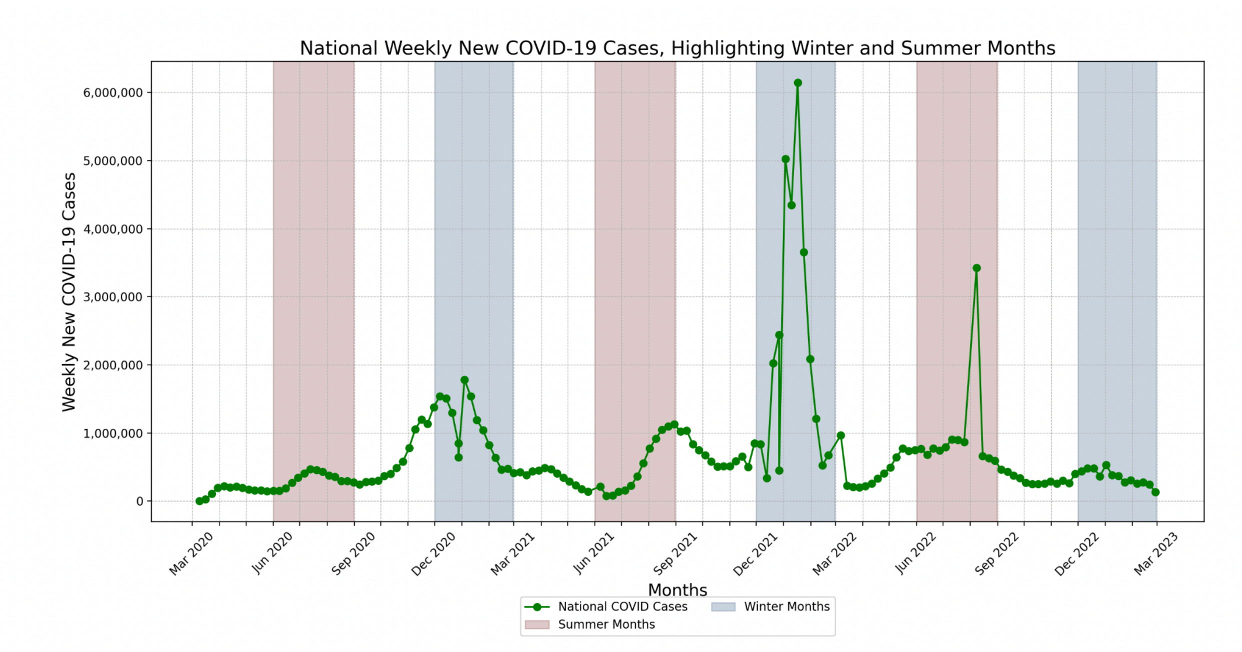
6. Visualizations Created (10 Points)

Visualization 1: Michigan Weekly New COVID-19 Cases, Highlighting Winter and Summer Mon



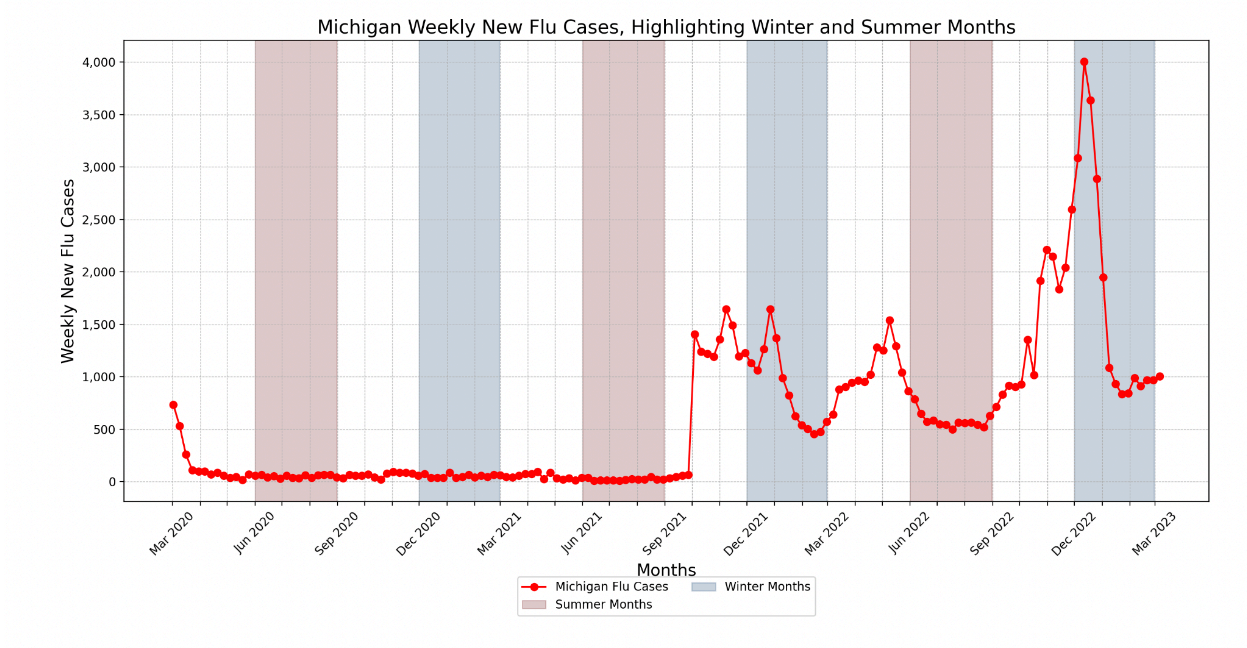
This chart shows the weekly new COVID-19 cases in Michigan,
with winter months highlighted in blue and summer months in red to highlight seasonal trends. In the Winter months of 2020 and 2021, there were initial slight and dramatic increases in the number of Michigan COVID-19 cases, respectively

Visualization 2: National Weekly New COVID-19 Cases, Highlighting Winter and Summer Months



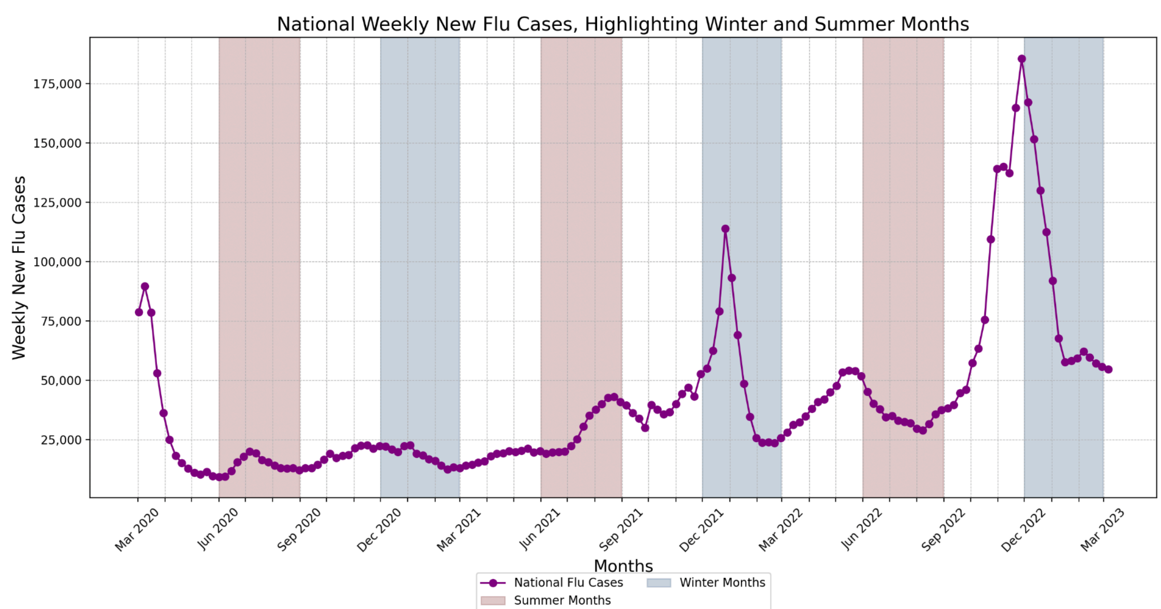
This chart shows the weekly new COVID-19 cases across the U.S.,
with winter months highlighted in blue and summer months in red to highlight seasonal trends. In the Winter months of 2020 and 2021, there were slight and dramatic increases in the number of national COVID-19 cases, respectively.

<u>Visualization 3</u>: Michigan Weekly New Flu Cases, Highlighting Winter and Summer Months



This chart displays the weekly new flu cases in the U.S., with winter months in blue and summer months in red to highlight seasonal trends. In the Winter months of 2021 and 2022, there were initial slight and dramatic increases in the number of Michigan flu cases, respectively.

<u>Visualization 4</u>: National Weekly New Flu Cases, Highlighting Winter and Summer Months



This chart displays the weekly new flu cases in the U.S., with winter months in blue and summer months in red to highlight seasonal trends. In the Winter months of 2021 and 2022, there were initial slight and dramatic increases in the number of national flu cases, respectively

7. Instructions for Running the Code (10 Points)

Setting Up the Environment:

Installing Dependencies: ensure you have the required Python libraries installed. You can create a virtual environment and install the necessary packages using the command:

```
pip install pandas matplotlib sqlite3 epiweeks requests meteostat
```

Import Necessary Libraries:

```
from meteostat import Point, Daily

from epiweeks import Week
```

Steps to Run the Code:

1. **Prepare the Database**:
   - Open the project folder in a Python editor (e.g. VScode)
2. **Run data_collection.py first**:
   - Before running main.py, you need to run data_collection.py to collect data from the APIs and insert it into the **SQLite database (final_project.db).**
   - The script will gather **25 rows of data per run** to meet the project's requirement of limiting data insertion.
   - **Run data_collection.py four times sequentially**:
     - After each run, the script will progressively populate the database with data from the APIs.
     - **Check the database**: After each run, ensure that 25 rows have been added. You can use a database browser (e.g. DB Browser for SQLite) to verify the data.
3. **Final Run for Data Collection**:
   - Run data_collection.py a fifth time.
   - This final run will gather the remaining rows of data, completing the data collection process and populating the database with the full dataset.
4. **Run main.py**:
   - Once you've run data_collection.py and populated the database, **run main.py**.
   - The main.py script will handle the entire workflow, ensuring data collection is completed and setting up for visualization generation
5. **Run data_visualization.py**
   - generate visualizations using the collected data.
   - create line charts for flu and COVID-19 trends, along with temperature correlations.
6. **Expected Outputs**:
   - **Database**: After the fifth run, the **SQLite database (final_project.db)** will contain tables for flu, COVID-19, and temperature data, populated with **over 100 rows** from each API.
   - **Visualizations**: The script will **generate line charts** for flu and COVID-19 trends, along with temperature correlations. These visualizations will be displayed as part of the script's output.

<u>Error Handling and Debugging</u>

Missing Modules:

- If you encounter the error ModuleNotFoundError, you may be missing required libraries. Ensure all dependencies are installed using:
  - pip install pandas matplotlib sqlite3 requests meteostat epiweeks

Data Duplication:

- If data appears duplicated in the database, make sure the run_count column is correctly tracking the number of runs. The script will prevent duplicates by ensuring that only new data is added each time.

8. Code Documentation (20 Points)

get_db_connection

- Purpose: Establishes and returns a connection to the SQLite database.
- Input Parameters: None
- Returns: A SQLite connection object (sqlite3.Connection) to interact with the database.
- Explanation: This function uses the sqlite3 module to create a connection to the final_project.db database, enabling SQL queries and data manipulation.

get_week_id(date)

- Purpose: Converts a date into a unique identifier for the year and week number.
- Input Parameters:
    - date (str): A date string in the format YYYY-MM-DD.
- Returns: An integer (int) representing the week ID (e.g., 202301 for the first week of 2023).
- Explanation: The function parses the input date using datetime.strptime and formats it to extract the year and week number using strftime.

process_weather_data(location, start_date, end_date, table_name)

- Purpose: Fetches daily weather data for a specified location and stores weekly aggregated data in the database.
- Input Parameters:
    - location (meteostat.Point): Geographical coordinates of the location.
    - start_date (datetime): Start date for weather data retrieval.
    - end_date (datetime): End date for weather data retrieval.
    - table_name (str): Name of the database table where processed data will be stored.
- Returns: None
- Explanation: Uses the Daily class from the meteostat package to fetch weather data, calculates weekly averages (temperature min, max, and avg), and stores the results in a specified database table.

process_all_cities()

- Purpose: Processes and stores weather data for predefined cities into respective database tables.
- Input Parameters: None
- Returns: None
- Explanation: Iterates through the predefined NATIONAL_LOCATIONS, calls process_weather_data for each city, and stores the results in individual city tables.

calculate_national_weather_average()

- Purpose: Calculates and stores national average temperatures (min, max, and avg) from city weather data.
- Input Parameters: None
- Returns: None
- Explanation: Combines weather data from multiple city tables, calculates the average temperature values, and stores the results in the national_weather table.

fetch_and_store_michigan_covid()

- Purpose: Fetches Michigan COVID-19 timeseries data and stores weekly aggregated data in the database.
- Input Parameters: None
- Returns: None
- Explanation: Uses the COVID Act Now API to retrieve Michigan COVID-19 data, processes daily cases into weekly totals, and stores them in the database.

fetch_and_store_national_covid()

- Purpose: Fetches national COVID-19 timeseries data and stores weekly aggregated data in the database.
- Input Parameters: None
- Returns: None
- Explanation: Similar to fetch_and_store_michigan_covid, but retrieves data for the entire U.S.

store_covid_data(data, table_name)

- Purpose: Processes raw COVID-19 data and saves weekly aggregated data to a specified table in the database.
- Input Parameters:
    - data (dict): JSON data containing COVID-19 timeseries.
    - table_name (str): Name of the database table to store processed data.
- Returns: None
- Explanation: Extracts daily cases from the raw data, calculates weekly totals, and stores the results in the database.

fetch_and_store_flu_data()

- Purpose: Fetches weekly flu data from the Delphi Epidata API and stores it in the database.
- Input Parameters: None
- Returns: None
- Explanation: Retrieves flu data for Michigan and the U.S., processes epiweeks into dates, and stores the data in the database.

collect_all_data()

- Purpose: Coordinates the collection of weather, COVID-19, and flu data.
- Input Parameters: None
- Returns: None
- Explanation: Sequentially calls functions to process weather data for Michigan and cities, compute national averages, and fetch/store COVID-19 and flu data.

format_with_commas(x, pos)

- Purpose: Formats numbers with commas for better readability in plots.
- Input Parameters:
  - x (float): The number to format.
  - pos (int): The tick position on the axis (used internally by Matplotlib).
- Returns: A string (str) with the number formatted with commas.
- Explanation: Used as a helper function for axis formatting in plots.

set_monthly_xticks(ax, start_date, end_date)

- Purpose: Sets custom x-axis ticks for a date range, labeling only specific months.
- Input Parameters:
  - ax (Axes): Matplotlib Axes object to modify.
  - start_date (str): Start of the date range (YYYY-MM-DD).
  - end_date (str): End of the date range (YYYY-MM-DD).
- Returns: None
- Explanation: Customizes x-axis labels for visual clarity in time-series plots.

plot_data(df, x_column, y_column, title, ylabel, label, color, start_date, end_date)

- Purpose: Creates a line plot of time-series data with formatted axes.
- Input Parameters:
  - df (DataFrame): DataFrame containing the data to plot.
  - x_column (str): Column for the x-axis.
  - y_column (str): Column for the y-axis.
  - title (str): Title of the plot.
  - ylabel (str): Label for the y-axis.
  - label (str): Legend label for the plot.
  - color (str): Color of the plot line.
  - start_date (str): Start date for the plot.
  - end_date (str): End date for the plot.
- Returns: None
- Explanation: Generates a simple time-series line plot with a formatted x-axis and labeled y-axis.

plot_cases_with_bars(df, x, y, label, color, title, ylabel, start_date, end_date, seasons)

- Purpose: Plots time-series data with seasonal highlight bars.
- Input Parameters:
  - df (DataFrame): DataFrame containing the data to plot.
  - x (str): Column for the x-axis.
  - y (str): Column for the y-axis.
  - label (str): Legend label for the line plot.
  - color (str): Line color.
  - title (str): Plot title.
  - ylabel (str): Label for the y-axis.
  - start_date (str): Start date for the x-axis.
  - end_date (str): End date for the x-axis.
  - seasons (list): List of tuples defining seasonal bars (start, end, label).
- Returns: None

- Explanation: Adds visual emphasis to specific seasonal periods on a time-series plot.

visualize_all_data()

- Purpose: Generates all visualizations, including COVID-19 and flu cases, with seasonal highlights.
- Input Parameters: None
- Returns: None
- Explanation: Iteratively generates plots for Michigan and national COVID-19 and flu data while incorporating seasonal highlights.

setup_database()

- Purpose: Ensures the database is correctly set up for the pipeline.
- Input Parameters: None
- Returns: Boolean (True if successful, False otherwise).
- Explanation: Verifies database connectivity and table existence.

collect_data()

- Purpose: Runs the data collection phase of the pipeline.
- Input Parameters: None
- Returns: Boolean (True if successful, False otherwise).
- Explanation: Orchestrates multiple data-fetching and processing functions.

create_visualizations()

- Purpose: Generates all visualizations in the analysis pipeline.
- Input Parameters: None
- Returns: Boolean (True if successful, False otherwise).
- Explanation: Calls the visualize_all_data function to generate insights from the collected data.

9. Documentation of Resources Used (20 Points)

Delphi Epidata API

- **Source:** FluView API
- **Purpose:** Collected weekly flu case data (num_ili, wili) for Michigan and nationally.

Covid Act Now API

- **Source:** COVID-19 API
- **Purpose:** Retrieved COVID-19 case data (daily confirmed cases) for Michigan and nationally.

Meteostat JSON API

- **Source:** Meteostat API
- **Purpose:** Gathered daily temperature data (`TAVG`, `TMIN`, `TMAX`) for Michigan.

Libraries Used:

- `pandas`: Data manipulation and analysis.
- `matplotlib`: Visualization creation.
- `sqlite3`: Database management.

10. Conclusion

The project successfully met its goals by collecting, processing, and analyzing data to investigate average temperatures and new flu and COVID-19 cases in Michigan and nationwide. The visualizations we created provided insights into how temperature fluctuations shifted with disease trends, achieving our original research objectives. Future expansions may include additional statistical tests to measure more specific correlations.