

Project Progress

Greenway: Locate Nearby Landfills and Transfer Stations

Ira A. Fulton Schools of Engineering

Arizona State University

IFT 593: Applied Project

Dr. Asmaa Elbadrawy

Due Date: 04/11/2024

Project Team:

Megha Jotangiya

Vrutik Adani

Harshita Mishra

Bhavya Atul Shah



Contents

Project Overview:	3
Scope and Objective:	5
Data Sources:	7
Data Cleaning:	10
Initial Data Preprocessing:	12
Code for Website:	17
Screenshots:	34
Tableau Visualization:	37
Insights & Actions:	44

Project Overview:

Sustainability is the practice of meeting current needs without compromising the ability of future generations to meet their own needs. This principle is crucial for several reasons, with one of the key aspects being environmental conservation. When we talk about sustainability, we're emphasizing the importance of using resources efficiently and reducing waste. This approach plays a significant role in preserving natural resources like clean air, water, forests, and biodiversity. By managing resources responsibly, we can protect ecosystems and minimize the impact of climate change.

The "Green Way: Locate Nearby Landfills & Transfer Stations" project contributes to sustainability in several ways:

1. Efficient Waste Management:

- The project aims to enhance waste management efficiency by providing users with a web application to locate nearby landfills and transfer stations within the Phoenix city area.
- Efficient waste management is crucial for sustainability as it reduces pollution, minimizes resource depletion, and prevents environmental degradation.
- By enabling users to easily find nearby waste facilities, the project promotes proper waste disposal, which contributes to preserving natural resources and ecosystems.
- Proper waste disposal also helps in reducing the release of harmful substances into the environment, thus safeguarding clean air, water, and soil for future generations.

2. Reduces Carbon Footprint:

- One of the key benefits of the "Green Way" project is its contribution to reducing carbon footprints.
- The project achieves this by facilitating easy access to waste facilities, which reduces the need for long-distance transportation of waste materials.
- By minimizing transportation distances, the project helps in lowering vehicle emissions, which are a major contributor to climate change.
- Lowering carbon emissions aids in climate change mitigation efforts and fosters environmental sustainability by curbing the rate of global warming and its associated impacts on ecosystems and human health.

3. Supports Green Infrastructure:

- The "Green Way" project supports the adoption of green infrastructure practices by providing digital tools that enhance waste management efficiency and promote sustainable urban development.
- Green infrastructure refers to natural or nature-based solutions for managing stormwater, reducing urban heat islands, improving air quality, and enhancing overall environmental quality.
- Through interactive maps and geolocation services, the project showcases the importance of green infrastructure in waste management and urban planning.
- By promoting green infrastructure adoption, the project encourages the use of sustainable practices that reduce resource consumption, improve resilience to environmental challenges, and enhance the quality of life for communities.

Scope and Objective:

The scope of the "Green Way" web application is to provide users with a convenient tool for locating nearby landfills and transfer stations within the Phoenix city area. The objective is to streamline waste disposal processes and promote sustainable waste management practices by offering users easy access to essential waste facilities.

Key Features and Functionalities:

1. User Address Validation:

- Users can input their address into the application.
- The application validates the entered address against an active_addresses.csv file to ensure accuracy and eligibility.
- Address validation helps prevent errors and ensures that users receive relevant information based on their location.

2. Location Display:

- Upon successful address validation, the application displays the entered address to users.
- It also provides service area information, indicating which landfills and transfer stations are within proximity to the user's address.
- Additionally, the application includes a link to the bulk-trash collection schedule, providing users with further waste management resources.
- The user's address is visually represented on a map using the OpenStreetMap API, enhancing the user experience and facilitating navigation.

3. Distance Calculation:

- The application utilizes the Mapbox Direction API to calculate the distance between the user's address and nearby landfills and transfer stations.
- Distance calculation helps determine the nearest station to the user, enabling efficient waste disposal planning.

4. Nearest Station Search:

- Users can initiate a search for the nearest landfill or transfer station by clicking on the designated button within the application.
- The application calculates the distance to each station based on the user's address and displays the nearest station prominently.
- It also shows the shortest driving route between the user's address and the selected station on the map, facilitating navigation.

5. Detailed Nearest Station Information:

- Upon selecting a station, the application provides users with detailed information about it.
- This includes the station's name, address, operating hours, holidays, and directions for reaching it.
- Additionally, the application displays the distance in miles from the user's address to the selected station, assisting users in making informed decisions about waste disposal options.

6. Error Handling:

- The application implements robust error handling mechanisms to address unexpected errors that may occur during address validation, map loading, or API interactions.
- Error handling ensures smooth functionality and enhances the user experience by minimizing disruptions and providing clear guidance in case of technical issues.

By incorporating these key features and functionalities, the "Green Way" web application aims to empower users with the tools and information necessary to make sustainable choices regarding waste disposal. It prioritizes user experience, accuracy, and reliability to promote efficient waste management practices and contribute to environmental sustainability in the Phoenix city area.

Data Sources:**1. Public_Works_Solid_Waste_Active_Service_Addresses.csv:**

- This dataset was obtained from the official Phoenix Open Data website at [link](#).
- It contains information about active service addresses related to solid waste management in the Phoenix city area.
- The dataset likely includes details such as addresses, service status, and other relevant information necessary for validating user input addresses within the application.

Column Name	Data Type	Description
service address	object	Address of the property
City	object	City name
State	object	State abbreviation
Zip	object	Zip code
INCYLIMIT	object	Indicator of city limits
REFUSE	object	Refuse collection service
RECYCLE	object	Recycling collection service
BULK_TRASH	object	Bulk trash collection service
GREEN_ORG	object	Green waste/organic waste collection service
ELIGIBLEGO	object	Eligibility for services
QUARTERSECTION	object	Quarter section
PROPERTY_TYPE	object	Property type
PREM_DESCRIPTION	object	Property description
NUMBER_OF_CONTAINERS	int	Number of waste containers
IS_IN_ALLEY	object	Indicator if property is in an alley
LIVINGUNITS	int	Number of living units
SERVICE_AREA	object	Service area or zone
GIS_X_COORDINATE	float	X-coordinate in GIS format
GIS_Y_COORDINATE	float	Y-coordinate in GIS format

Data Dictionary: Public_Works_Solid_Waste_Active_Service_Addresses.csv

2. Transfer_Station_Dataframe.csv:

- This file was created by the project team and contains data specific to landfills and transfer stations within the Phoenix city area.
 - It serves as a reference for the locations and details of these waste facilities, including their names, addresses, and possibly operational hours or other relevant information.
 - The data from this file is utilized by the application to provide users with information about nearby waste disposal options.

Column Name	Description
Name	Name of the location or station
Station_Address	Address of the station
Latitude	Latitude coordinates of the location
Longitude	Longitude coordinates of the location

Data Dictionary: Transfer_Station_Dataframe.csv

3. City_Limit_Dark_Outline.geojson:

- The project utilizes the "City_Limit_Dark_Outline.geojson" file sourced from the Phoenix Open Data portal at [link](#).
 - This file contains geographical data representing the boundary of the Phoenix city area.
 - It is used to visualize the extent of the city area within the application, providing users with context regarding the coverage area for waste management services.
 - The boundary data helps users understand which waste facilities fall within the jurisdiction of Phoenix and are accessible through the application.

Data: City Limit Dark Outline.geojson

4. Df23:

- This dataset has been used for Tableau Visualizations.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	Id	SiteCode	MT	MTLabel	MTGroup	DateOut	NetSTN	BillCustomerType														
2	0	1	27	100 Refuse	REFUSE	#####	2.81	AR														
3	1	2	27	100 Refuse	REFUSE	#####	2.84	Other Dept														
4	2	3	27	100 Refuse	REFUSE	#####	3.09	AR														
5	3	4	27	100 Refuse	REFUSE	#####	7.56	Contained														
6	4	5	27	100 Refuse	REFUSE	#####	1.54	AR														
7	5	6	27	100 Refuse	REFUSE	#####	9.5	Contained														
8	6	7	27	200 Recycle	RECYCLE	#####	1.42	Reciprocal														
9	7	8	27	100 Refuse	REFUSE	#####	3.11	AR														
10	8	9	27	100 Refuse	REFUSE	#####	5.94	Other Dept														
11	9	10	27	300 Green	GREEN	#####	2.34	Contained														
12	10	11	27	100 Refuse	REFUSE	#####	7.52	Uncontained														
13	11	12	27	100 Refuse	REFUSE	#####	0.98	AR														
14	12	13	27	300 Green	GREEN	#####	3.84	AR														
15	13	14 NG	27	100 Refuse	REFUSE	#####	0.28	AR														
16	14	15	27	200 Recycle	RECYCLE	#####	0.84	Reciprocal														
17	15	16 NG	27	100 Refuse	REFUSE	#####	0.66	AR														
18	16	17	27	200 Recycle	RECYCLE	#####	4.05	Contained														
19	17	18	27	100 Refuse	REFUSE	#####	8.53	Uncontained														
20	18	19	27	200 Recycle	RECYCLE	#####	3.39	Contained														
21	19	20 NG	27	200 Recycle	RECYCLE	#####	3.09	Contained														
22	20	21	27	100 Refuse	REFUSE	#####	1.47	AR														
23	21	22 NG	27	100 Refuse	REFUSE	#####	3.31	AR														
24	22	23	27	200 Recycle	RECYCLE	#####	4.93	Contained														
25	23	24 NG	27	200 Recycle	RECYCLE	#####	1.13	Reciprocal														
26	24	25	27	100 Refuse	REFUSE	#####	7.14	Uncontained														
27	25	26	27	200 Recycle	RECYCLE	#####	4.14	Contained														
28	26	27	27	200 Recycle	RECYCLE	#####	6.22	Contained														
29	27	28	27	200 Recycle	RECYCLE	#####	3.13	Contained														

Contents of df23.csv

By leveraging these data sources, the "Green Way" web application is able to provide users with accurate and relevant information about waste management services within the Phoenix city area. The combination of address validation, facility data, and geographical boundaries enhances the functionality and usefulness of the application, ultimately supporting the project's objective of promoting sustainable waste management practices.

Data Cleaning:

```
In [67]: import pandas as pd
df=pd.read_csv("Public_Works_Solid_Waste_Active_Service_Addresses.csv")

In [68]: df.head()
```

	_id	service address	City	State	Zip	INCITYLIMIT	REFUSE	RECYCLE	BULK_TRASH	ELIGIBLEGO	QUARTERSECTION	PROPERTY_TYPE
0	1	3826520535	8415 N 31ST LN	PHOENIX AZ	85051-6011	Y	A4100	A4200	BTW11	Y	25-21	R1
1	2	3826530549	4913 W JOHNSON CIR	GLENDALE AZ	85308-3406	Y	A6100	A6200	BTW01	Y	37-17	R1
2	3	3826600960	2916 S 94TH GLEN	TOLLESON AZ	85353-0000	Y	G4100	G4200	BTS01	N	6-6	R1
3	4	3826610723	5521 W AVALON DR	PHOENIX AZ	85031-3312	Y	A2100	A2200	BTW09	N	15-15	R1
4	5	3826620440	3401 E DESERT COVE AVE	PHOENIX AZ	85028-2714	Y	D2100	D2200	BTN08	Y	29-35	R1

5 rows × 21 columns

```
In [69]: df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 391043 entries, 0 to 391042
Data columns (total 21 columns):
 #   Column          Non-Null Count  Dtype  
 --- 
 0   _id             391043 non-null   int64  
 1   id              391043 non-null   int64  

```

```
In [69]: df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 391043 entries, 0 to 391042
Data columns (total 21 columns):
 #   Column          Non-Null Count  Dtype  
 --- 
 0   _id             391043 non-null   int64  
 1   id              391043 non-null   int64  

```

```
In [70]: df.isnull().sum()
Out[70]:
_id                0
service address    0
City               0
State              0
Zip                0
INCITYLIMIT        0
REFUSE             573
RECYCLE            12779
BULK_TRASH         5513
GREEN_ORG          388406
ELIGIBLEGO         0
QUARTERSECTION    11
PROPERTY_TYPE      2
PREM_DESCRIPTION   2
NUMBER_OF_CONTAINERS 0
IS_IN_ALLEY        0
LIVINGUNITS        676
SERVICE_AREA       24111
GIS_X_COORDINATE  24111
GIS_Y_COORDINATE  24111
dtype: int64
```

IFT 593 Final Project

The screenshot shows a Jupyter Notebook running on a local host. The notebook has three cells:

- In [71]:** Contains Python code to drop columns, rename columns, and display the first few rows of the DataFrame.
- Out[71]:** Displays the first five rows of the DataFrame, showing columns: Id, Service_Address, City, State, Zip, Number_of_Containers, and Service_Area.
- In [72]:** Contains Python code to check for null values in the DataFrame.
- Out[72]:** Displays the result of the null check, showing all counts are 0.
- In [73]:** Contains Python code to capitalize words in the 'Service_Address' column.
- Out[73]:** Displays the first few rows of the DataFrame after capitalization.

The screenshot shows a Jupyter Notebook running on a local host. The notebook has two cells:

- In [73]:** Contains Python code to capitalize words in the 'Service_Address' column, title case the 'City' column, remove zip codes from the 'Zip' column, merge columns, and clean the 'Address' column.
- Out[73]:** Displays the first few rows of the DataFrame, showing columns: Id, Number_of_Containers, Service_Area, and Address.
- In [74]:** Contains Python code to save the DataFrame as a CSV file.
- In []:** An empty cell for future input.

Initial Data Preprocessing:

jupyter Applied_Project_Phoenix_Sustainability_Dashboard Last Checkpoint: 02/29/2024 (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted Python 3 (pykernel) ○

PART I : DATA CLEANING AND PRE-PROCESSING

PUBLIC WORKS SOLID WASTE ACTIVE SERVICES ADDRESSES

Column Name	Data Type	Description
service_address	object	Address of the property
City	object	City name
State	object	State abbreviation
Zip	object	Zip code
INCITYLIMIT	object	Indicator of city limits
REFUSE	object	Refuse collection service
RECYCLE	object	Recycling collection service
BULK_TRASH	object	Bulk trash collection service
GREEN_ORG	object	Green waste/organic waste collection service
ELIGIBLEOG	object	Eligibility for services
QUARTERSECTION	object	Quarter section
PROPERTY_TYPE	object	Property type
PREM_DESCRIPTION	object	Property description
NUMBER_OF_CONTAINERS	int	Number of waste containers
IS_IN_ALLEY	object	Indicator if property is in an alley
LIVINGUNITS	int	Number of living units
SERVICE_AREA	object	Service area or zone
GIS_X_COORDINATE	float	X-coordinate in GIS format
GIS_Y_COORDINATE	float	Y-coordinate in GIS format

78°F Mostly sunny Search ENG IN 6:20 PM 4/11/2024

jupyter Applied_Project_Phoenix_Sustainability_Dashboard Last Checkpoint: 02/29/2024 (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted Python 3 (pykernel) ○

```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [2]: # Read the CSV file into a DataFrame
data = pd.read_csv('/Users/harshitamishra/Downloads/Public_Works_Solid_Waste_Active_service_addresses.csv')
```

```
In [3]: # View the first few rows of the DataFrame
data.head() # View the first few rows of the DataFrame
data.info() # Get information about the DataFrame, including data types and missing values
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 391043 entries, 0 to 391042
Data columns (total 21 columns):
 #   Column           Non-Null Count  Dtype  
 --- 
 0   _id              391043 non-null   int64  
 1   id               391043 non-null   int64  
 2   service_address  391043 non-null   object  
 3   City              391043 non-null   object  
 4   State             391043 non-null   object  
 5   Zip               391043 non-null   object  
 6   INCITYLIMIT       391043 non-null   object  
 7   REFUSE            390570 non-null   object  
 8   RECYCLE           379337 non-null   object  
 9   BULK_TRASH         385530 non-null   object  
 10  GREEN_ORG          10638 non-null   object  
 11  ELIGIBLEOG        391043 non-null   object  
 12  QUARTERSECTION    391032 non-null   object  
 13  PROPERTY_TYPE     391041 non-null   object  
 14  PREM_DESCRIPTION   391041 non-null   object  
 15  NUMBER_OF_CONTAINERS 391043 non-null   int64  
 16  IS_IN_ALLEY        391043 non-null   object  
 17  LIVINGUNITS        391043 non-null   int64  
 18  SERVICE_AREA        300367 non-null   object
```

78°F Mostly sunny Search ENG IN 6:21 PM 4/11/2024

IFT 593 Final Project

jupyter Applied_Project_Phoenix_Sustainability_Dashboard Last Checkpoint: 02/29/2024 (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel) | Logout

	391042	391043	1310810358	4236 N 85TH AVE	PHOENIX	AZ	85037-2109	Y	G3100	G3200	BTW07	Y	17-8
391043 rows × 20 columns													

```
In [6]: # Get information about the DataFrame, including data types and missing values
data.head() # View the first few rows of the DataFrame
data.info()
```

#	Column	Non-Null Count	Dtype
0	_id	391043	non-null int64
1	Id	391043	non-null int64
2	service_address	391043	non-null object
3	City	391043	non-null object
4	State	391043	non-null object
5	Zip	391043	non-null object
6	INCITYLIMIT	391043	non-null object
7	REFUSE	390470	non-null object
8	BULCYCLE	372553	non-null object
9	BULCTRASH	365539	non-null object
10	ELIGIBLEGO	391043	non-null object
11	QUARTERSECTION	391032	non-null object
12	PROPERTY_TYPE	391041	non-null object
13	PREM_DESCRIPTION	391043	non-null object
14	NUMBER_OF_CONTAINERS	391043	non-null int64
15	IS_IN_ALLEY	391043	non-null object
16	LIVINGUNITS	391043	non-null int64
17	SERVICE_AREA	390367	non-null object
18	GIS_X_COORDINATE	366932	non-null float64
19	GIS_Y_COORDINATE	366932	non-null float64

dtypes: float64(2), int64(4), object(14)
memory usage: 59.7+ MB

IFT 593 Final Project

The screenshot shows a Jupyter Notebook running on a local host. The notebook has several cells:

- In [7]:

```
# Filter the DataFrame to include only rows where the 'City' column is 'PHOENIX'  
phoenix_data = data[data['City'] == "PHOENIX"]
```
- In [8]:

```
phoenix_data.shape # Display the shape of the filtered DataFrame
```
- Out[8]:

```
(326534, 20)
```
- In [9]:

```
phoenix_data.isnull().sum() # Check for missing values in the filtered DataFrame
```
- Out[9]:

Column	Count
_id	0
id	0
service address	0
City	0
State	0
ZIP	0
DICITYLIMIT	0
REFUSE	513
RECYCLE	12681
BULK_TRASH	5348
ELIGIBLEO	0
QUARTERSECTION	10
PROPERTY_TYPE	2
PREH_DESCRIPTION	1
NUMBER_OF_CONTAINERS	0
IS_IN_ALLEY	0
LIVINGUNITS	0
SERVICE_AREA	631
GIS_X_COORDINATE	17439
GIS_Y_COORDINATE	17439
- In [10]:

```
# Calculate the percentage of missing values for each column  
missing_percentage = (data.isnull().sum() / len(data)) * 100
```
- # Display the percentage of missing values
print("Percentage of Missing Values:\n", missing_percentage)

The status bar at the bottom indicates it's 6:22 PM on 4/11/2024, the weather is 78°F mostly sunny, and the system language is ENG IN.

The screenshot shows a Jupyter Notebook running on a local host. The notebook has several cells:

- In [10]:

```
# Calculate the percentage of missing values for each column  
missing_percentage = (data.isnull().sum() / len(data)) * 100
```
- # Display the percentage of missing values
print("Percentage of Missing Values:\n", missing_percentage)
- Percentage of Missing Values:

Column	Percentage
_id	0.000000
id	0.000000
service address	0.000000
City	0.000000
State	0.000000
ZIP	0.000000
DICITYLIMIT	0.000000
REFUSE	0.146531
RECYCLE	3.268183
BULK_TRASH	1.409919
ELIGIBLEO	0.000000
QUARTERSECTION	0.002813
PROPERTY_TYPE	0.000511
PREH_DESCRIPTION	0.000511
NUMBER_OF_CONTAINERS	0.000000
IS_IN_ALLEY	0.000000
LIVINGUNITS	0.000000
SERVICE_AREA	0.172871
GIS_X_COORDINATE	6.165818
GIS_Y_COORDINATE	6.165818
- In [11]:

```
phoenix_data.dropna(inplace=True) # Drop rows with missing values from the filtered DataFrame
```
- /var/folders/5v/xtgmy_dx10d8y3wv9_g_pzndw0000gn/T/ipykernel_4705/2345750581.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
phoenix_data.dropna(inplace=True) # Drop rows with missing values from the filtered DataFrame

The status bar at the bottom indicates it's 6:22 PM on 4/11/2024, the weather is 78°F mostly sunny, and the system language is ENG IN.

IFT 593 Final Project

(28) WhatsApp x Meet - fku-pdpx x GREEN WAY - Present x Final Project Delivery x Home Page - Select x Applied_Project_Phox x Green Way; Locate x

localhost:8889/notebooks/Applied_Project_Phox_Sustainability_Dashboard.ipynb

jupyter Applied_Project_Phox_Sustainability_Dashboard Last Checkpoint: 02/29/2024 (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Logout Not Trusted Python 3 (ipykernel)

In [12]: `phoenix_data # Check for missing values again after dropping rows`

Out[12]:

_id	id	service_address	City	State	Zip	INCITYLIMIT	REFUSE	RECYCLE	BULK_TRASH	ELIGIBLEGO	QUARTERSECTION	PROPERTY	
0	1	8415 N 31ST LN	PHOENIX	AZ	85051-5011	Y	A4100	A4200	BTW11	Y	25-21		
3	4	5521 W AVIA DR	PHOENIX	AZ	85031-3312	Y	A2100	A2200	BTW09	N	15-15		
4	5	3401 E DESERT COVE AVE	PHOENIX	AZ	85028-2714	Y	D2100	D2200	BTN08	Y	29-35		
6	7	1007 W ORAIBI DR	PHOENIX	AZ	85027-4649	Y	C5100	C5200	BTN01	N	40-26		
7	8	11023 N 32ND AVE	PHOENIX	AZ	85029-4156	Y	A4100	A4200	BTW11	Y	29-21		
...		
391035	391036	1310646764	3730 E ZACH DR	PHOENIX	AZ	85050-0000	Y	C3100	C3200	BTN04	Y	42-36	
391039	391040	1310730492	401 E SEQUOIA DR	PHOENIX	AZ	85024-1624	Y	C5100	C5200	BTN03	N	40-28	
391040	391041	1310742704	5440 W FETTUOK TH	PHOENIX	AZ	85053-0000	Y	C5100	C6200	BTN13	Y	49-16	
391041	391042	1310800322	2821 W GARFIELD ST	PHOENIX	AZ	85009-3926	Y	G2100	G2200	BT510	Y	22-Nov	
391042	391043	1310810358	4236 N 85TH AVE	PHOENIX	AZ	85037-2109	Y	G3100	G3200	BTW07	Y	17-8	

293593 rows × 20 columns

(28) WhatsApp X Meet - fku-pdk! GREEN WAY - Present Final Project Delivery Home Page - Select Applied_Project_Ph... Green Way: Locate ...

localhost:8889/notebooks/Applied_Project_Phoenix_Sustainability_Dashboard.ipynb

jupyter Applied_Project_Phoenix_Sustainability_Dashboard Last Checkpoint: 02/29/2024 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted Python 3 (ipykernel)

In [13]: `phoenix_data.isnull().sum() # Check for missing values in the filtered DataFrame`

Out[13]:

_id	0
id	0
service_address	0
City	0
State	0
Zip	0
INCITYLIMIT	0
REFUSE	0
RECYCLE	0
BULK_TRASH	0
ELIGIBLEGO	0
QUARTERSECTION	0
PROPERTY_TYPE	0
PREM_DESCRIPTION	0
NUMBER_OF_CONTAINERS	0
IS_IN_ALLEY	0
LIVESTOCKS	0
SERVICE_AREA	0
GIS_X_COORDINATE	0
GIS_Y_COORDINATE	0

dtype: int64

In [14]: `duplicate_id = phoenix_data[phoenix_data.duplicated(subset='id', keep=False)]`

In [15]: `duplicate_id # Display duplicate rows based on the 'id' column`

Out[15]:

_id	id	service_address	City	State	Zip	INCITYLIMIT	REFUSE	RECYCLE	BULK_TRASH	ELIGIBLEGO	QUARTERSECTION	PROPERTY_TYPE	PREM_DESCRIPTION
1	1	1	1	1	1	1	1	1	1	1	1	1	1

In [16]: `phoenix_data.dtypes # Display data types of columns in the filtered DataFrame`

Out[16]:

_id	int64
id	int64
service_address	object

IFT 593 Final Project

The screenshot shows a Jupyter Notebook running on a local host. The notebook title is "Applied_Project_Phoenix_Sustainability_Dashboard.ipynb". The code in cell [17] is:

```
In [17]: # Display the top 5 most common values in the 'PREM_DESCRIPTION' column
phoenix_data['PREM_DESCRIPTION'].value_counts().head(5)
```

The output of cell [17] is:

```
Out[17]: PREM_DESCRIPTION
SFR    287233
TFR     3268
TRFR    1884
TSFR     881
APTS     438
Name: count, dtype: int64
```

The code in cell [18] is:

```
In [18]: # Drop specified columns from the filtered DataFrame
phoenix_data.drop(columns = ['_id','City','State','ELIGIBLEGO','QUARTERSECTION'], inplace=True)
```

A warning message is displayed:

```
/var/folders/5v/x7gn_y_dx10dby3wvq9_pmdw0000gn/T/ipykernel_14705/2246369823.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

The code in cell [19] is:

```
In [19]: # Display the filtered DataFrame after dropping specified columns
phoenix_data
```

The output of cell [19] is a table:

	<u>_id</u>	<u>service_address</u>	<u>Zip</u>	<u>INCITYLIMIT</u>	<u>REFUSE</u>	<u>RECYCLE</u>	<u>BULK_TRASH</u>	<u>PROPERTY_TYPE</u>	<u>PREM_DESCRIPTION</u>	<u>NUMBER_OF_CONTAINERS</u>	
0	3826520535	8415 N 85051-6011			Y	A4100	A4200	BTW11	RES	SFR	2
3	3826610723	5521 W AVALON DR 85051-3312			Y	A2100	A2200	BTW09	RES	SFR	3
4	3826620440	3401 E COVE AVE 85028-2714			Y	D2100	D2200	BTN08	RES	SFR	3
6	3826630536	1007 W CRAIB DR 85027-4649			Y	C5100	C5200	BTN01	RES	SFR	2
7	3826720073	11023 N 85029-4158			Y	A4100	A4200	BTW11	RES	SFR	2
...
391035	1310646764	3730 E ZACHARY DR 85050-0000			Y	C3100	C3200	BTN04	RES	SFR	2
391039	1310730492	401 E SEQUOIA DR 85024-1624			Y	C5100	C5200	BTN03	RES	SFR	2
391040	1310742714	5440 W FETTERICK 85083-.....			Y	C6100	C6200	BTN13	RFS	SFR	?

The screenshot shows a Jupyter Notebook running on a local host. The notebook title is "Applied_Project_Phoenix_Sustainability_Dashboard.ipynb". The code in cell [18] is:

```
In [18]: # Drop specified columns from the filtered DataFrame
phoenix_data.drop(columns = ['_id','City','State','ELIGIBLEGO','QUARTERSECTION'], inplace=True)
```

A warning message is displayed:

```
/var/folders/5v/x7gn_y_dx10dby3wvq9_pmdw0000gn/T/ipykernel_14705/2246369823.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

The code in cell [19] is:

```
In [19]: # Display the filtered DataFrame after dropping specified columns
phoenix_data
```

The output of cell [19] is a table:

	<u>_id</u>	<u>service_address</u>	<u>Zip</u>	<u>INCITYLIMIT</u>	<u>REFUSE</u>	<u>RECYCLE</u>	<u>BULK_TRASH</u>	<u>PROPERTY_TYPE</u>	<u>PREM_DESCRIPTION</u>	<u>NUMBER_OF_CONTAINERS</u>	
0	3826520535	8415 N 85051-6011			Y	A4100	A4200	BTW11	RES	SFR	2
3	3826610723	5521 W AVALON DR 85051-3312			Y	A2100	A2200	BTW09	RES	SFR	3
4	3826620440	3401 E COVE AVE 85028-2714			Y	D2100	D2200	BTN08	RES	SFR	3
6	3826630536	1007 W CRAIB DR 85027-4649			Y	C5100	C5200	BTN01	RES	SFR	2
7	3826720073	11023 N 85029-4158			Y	A4100	A4200	BTW11	RES	SFR	2
...
391035	1310646764	3730 E ZACHARY DR 85050-0000			Y	C3100	C3200	BTN04	RES	SFR	2
391039	1310730492	401 E SEQUOIA DR 85024-1624			Y	C5100	C5200	BTN03	RES	SFR	2
391040	1310742714	5440 W FETTERICK 85083-.....			Y	C6100	C6200	BTN13	RFS	SFR	?

Code for Website:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Green Way: Locate Nearby Landfills & Transfer Stations</title>
<link rel="stylesheet" href="https://unpkg.com/leaflet/dist/leaflet.css" />
<script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>
<style>
body, html {
margin: 0;
padding: 0;
height: 100%;
}
.logo-title-container {
display: flex;
align-items: center;
padding: 10px;
background-color: #355E3B;
color: white;
}
.logo-title-container img {
width: 40px; /* Adjust the width of the logo */
margin-right: 10px; /* Adjust margin as needed */
}
h1 {
margin: 0;
font-size: 24px;
```

}

```
#mapContainer {
```

```
    width: 75%;
```

```
    height: auto;
```

```
    position: absolute;
```

```
    top: 0;
```

```
    left: 0;
```

```
    right: 0;
```

```
    bottom: 0;
```

}

```
#addressInput {
```

```
    width: 100%;
```

```
    padding: 10px;
```

```
    margin-bottom: 10px;
```

```
    border: 2px solid green;
```

```
    border-radius: 4px;
```

```
    box-sizing: border-box;
```

}

```
.white-box {
```

```
    width: 70%; /* Set the width to 70% */
```

```
    max-width: 360px; /* Adjust the max-width as needed */
```

```
    background-color: white;
```

```
    padding: 15px;
```

```
    box-sizing: border-box;
```

```
    z-index: 999;
```

```
    position: absolute;
```

```
    top: 0; /* Adjust top position as needed */
```

```
right: 0; /* Adjust right position as needed */  
border-radius: 0; /* Rounded corners */  
overflow: auto;  
}  
.transfer-details-box {  
display: none;  
margin-top: 5px;  
background-color: #f0f0f0; /* Light grey background color */  
padding: 10px; /* Optional: Add padding to the result box */  
border-radius: 5px; /* Optional: Add rounded corners */  
}  
.arrow-submit,  
.additional-btn {  
width: 100%;  
padding: 8px;  
margin-bottom: 10px;  
background-color: #355E3B;  
color: white;  
border: none; /* Remove border */  
border-radius: 4px;  
cursor: pointer;  
transition: background-color 0.3s ease;  
box-shadow: 2px 2px 4px rgba(0, 0, 0, 0.2); /* Add box shadow */  
/* Additional styles for h6 appearance */  
font-size: 1em;  
font-weight: bold;  
text-align: center;  
font-family: Vivaldi;  
}
```

```
.search-container {  
    display: flex;  
    flex-direction: column; /* Stack elements vertically */  
    align-items: center;  
    justify-content: center;  
  
}  
  
.result-box {  
    display: none; /* Initially hide the result box */  
    margin-top: 5px;  
    background-color: #f0f0f0; /* Light grey background color */  
    padding: 10px; /* Optional: Add padding to the result box */  
    border-radius: 5px; /* Optional: Add rounded corners */  
    margin-bottom: 10px;  
}  
  
#result-box p {  
    margin-bottom: 0;  
}  
  
#validationResult {  
    margin-top: 10px;  
    margin-bottom: 10px;  
    display: none; /* Hide the element by default */  
}  
  
#enteredAddress,  
#stationName {  
    color: #355E3B;  
    font-weight: bold;  
}  
  
.image-title-container {
```

```
display: flex;
align-items: center;
}

.image-title-container img {
    margin-right: 10px; /* Adjust the margin as needed */
}

.station-distance {
    color: grey;
    margin-left: 130px;
    font-weight: lighter;
}

</style>

</head>

<body>

<div class="white-box" id="whiteBox">

    <!-- Content inside the white box -->

    <div class="search-container">

        <div class="image-title-container">

            <h3 style="font-size: 23px; margin-top: 0; margin-bottom: 5px; color: #355E3B;">Find Drop-Off Location Near You</h3>

        </div>

        <input type="text" id="addressInput" name="addressInput" placeholder="3401 E Desert Cove Ave Phoenix AZ 85028" required>

        <p id="validationResult" style="display: none;"></p>

        <button class="arrow-submit" onclick="validateAddress()">Show Details</button>

    </div>

    <div class="result-box" id="resultSet">
```

```
<p id="enteredAddress" style="margin-top: 0; font-size: 16px;"></p>
<p id="serviceArea" style="margin-top: -10px;"></p>
<p id="collectionSchedule" style="margin-top: -10px; margin-bottom: 0;"></p>
</div>

<button    class="additional-btn"    onclick="showNearestStation()">Landfill/Transfer
Station</button>

<!-- Transfer station details section -->
<div class="transfer-details-box" id="transferStationDetails">
<p id="stationName" style="margin-top: 0; font-size: 16px;"></p>
<p id="stationDistance" style="margin-top: 0; color: grey; margin-left: 30px;"></p>
<p id="stationAddress" style="margin-top: -10px; "></p>
<h5 style="font-family: Vivaldi; margin-top: 0; margin-bottom: 0; font-size: 16px; font-
weight: bold;">Hours</h5>
<p id="stationHours" style="margin-top: 0; color: gray; "></p> <!-- Add this line for
displaying hours -->
<h5 style="font-family: Vivaldi; margin-top: 0; margin-bottom: 0; font-size: 16px; font-
weight: bold;">Holidays</h5>
<p id="stationHolidays" style="margin-top: 0; color: gray;"></p>
<h5 style="font-family: Vivaldi; margin-top: 0; margin-bottom: 0; font-size: 16px; font-
weight: bold;">Direction</h5>
<p id="stationDirection" style="margin-top: 0; margin-bottom: 0; color: gray;"></p>
</div>
</div>

<div id="mapContainer" class="map-container"></div>

<script src="https://unpkg.com/leaflet/dist/leaflet.js"></script>
<script src='https://api.mapbox.com/mapbox-gl-js/v2.6.1/mapbox-gl.js'></script>
<link href='https://api.mapbox.com/mapbox-gl-js/v2.6.1/mapbox-gl.css' rel='stylesheet' />

<script>
```

```
// Create new map with marker and tooltip
let map = L.map('mapContainer').setView([33.5987844812527,-112.077087523346], 10);

// Add a tile layer (OpenStreetMap tiles)
L.tileLayer('https://s.tile.openstreetmap.org/{z}/{x}/{y}.png', {attribution: '&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors' }).addTo(map);

// Load and add the GeoJSON boundary data to the map
var geojsonUrl = 'http://127.0.0.1:8080/City_Limit_Dark_Outline.geojson'; // Update the URL with your GeoJSON file URL
fetch(geojsonUrl)
  .then(response => response.json())
  .then(data => {
    L.geoJSON(data, {
      style: {
        color: 'green', // Adjust color if needed
        weight: 2, // Adjust line thickness if needed
        fillOpacity: 0,
        fillColor: 'none'
      }
    }).addTo(map);
  })
  // Fit the map to the GeoJSON boundary layer
  .catch(error => {
    console.error('Error fetching GeoJSON:', error);
  });
})
```

```
let marker = null;  
let stations = [  
  {  
    name: 'North Gateway Station',  
    address: '30205 N Black Canyon Hwy \n Phoenix AZ 85085',  
    lat: 33.7605925,  
    lon: -112.1163813,  
    hours: {  
      mondayToFriday: '5:30 a.m. to 5 p.m.',  
      saturday: '6 a.m. to 3 p.m.',  
      sunday: 'Closed'  
    },  
    holidays: [  
      "Martin Luther King Jr. Day",  
      "President's Day",  
      "Cesar Chavez Day",  
      "Memorial Day",  
      "Juneteenth",  
      "Independence Day",  
      "Labor Day",  
      "Indigenous Peoples' Day",  
      "Veterans Day (observed by the City on November 10, 2023)",  
      "Thanksgiving and the day after",  
      "Christmas",  
      "New Year's",  
      "Christmas Eve - Open 6 a.m. - 12 p.m.",  
      "Regular hours on New Year's Eve"  
    ]  
}
```

```
},  
{  
    name: '27th Ave Station',  
    address: '3060 S 27th Ave \n Phoenix AZ 85009',  
    lat: 33.4169941,  
    lon: -112.1186168,  
    hours: {  
        mondayToFriday: '5:30 a.m. to 5 p.m.',  
        saturday: '6 a.m. to 3 p.m.',  
        sunday: 'Closed'  
    },  
    holidays: [  
        "Martin Luther King Jr. Day",  
        "President's Day",  
        "Cesar Chavez Day",  
        "Memorial Day",  
        "Juneteenth",  
        "Independence Day",  
        "Labor Day",  
        "Indigenous Peoples' Day",  
        "Veterans Day (observed by the City on November 10, 2023)",  
        "Thanksgiving and the day after",  
        "Christmas",  
        "New Year's",  
        "Christmas Eve - Open 6 a.m. - 12 p.m.",  
        "Regular hours on New Year's Eve"  
    ]  
},  
{
```

```
name: '85 Landfill',
address: '28361 W Patterson Road \n Buckeye AZ 85326',
lat: 33.1890375,
lon: -112.6686124,
hours: {
    mondayToFriday: '5:30 a.m. to 5 p.m.',
    saturday: '6 a.m. to 3 p.m.',
    sunday: 'Closed'
},
holidays: [
    "Martin Luther King Jr. Day",
    "President's Day",
    "Cesar Chavez Day",
    "Memorial Day",
    "Juneteenth",
    "Independence Day",
    "Labor Day",
    "Indigenous Peoples' Day",
    "Veterans Day (observed by the City on November 10, 2023)",
    "Thanksgiving and the day after",
    "Christmas",
    "New Year's",
    "Christmas Eve - Open 6 a.m. - 12 p.m.",
    "Regular hours on New Year's Eve"
]
};

async function validateAddress() {
```

```
const addressInput = document.getElementById('addressInput').value.trim().toLowerCase();  
  
try {  
    const response = await fetch('http://127.0.0.1:8080/active-service-addresses.csv');  
    if (!response.ok) {  
        throw new Error('Failed to fetch CSV data.');//  
    }  
    const csvData = await response.text();  
    const rows = csvData.split('\\n');//  
    const headers = rows[0].split(',').map(header => header.trim()); // Trim whitespace from  
    headers  
    const addressIndex = headers.indexOf('Address');//  
    const serviceAreaIndex = headers.indexOf('Service _ Area');//  
  
    let isValid = false;  
    let validatedAddress = '';  
    let serviceArea = '';  
  
    for (let i = 1; i < rows.length; i++) {  
        const rowData = rows[i].split(',').map(item => item.trim()); // Trim whitespace from  
        row data  
        const address = rowData[addressIndex].toLowerCase();  
  
        if (address === addressInput) {  
            isValid = true;  
            validatedAddress = address;  
            serviceArea = rowData[serviceAreaIndex];  
  
            document.getElementById('resultBox').style.display = 'block'; // Show the result box  
        }  
    }  
}
```

```
const validationResult = document.getElementById('validationResult');

const enteredAddress = document.getElementById('enteredAddress');

enteredAddress.innerText = rowData[addressIndex]; // Display entered address

document.getElementById('serviceArea').innerHTML = 'Service Area: ' +
serviceArea; // Display service area

document.getElementById('collectionSchedule').innerHTML = '<a href="https://www.phoenix.gov/publicworkssite/Documents/2024%20Bulk%20Trash%20Collection%20Schedule_English.pdf" target="_blank">Bulk-Trash Collection Day</a>';

// Fetch latitude and longitude using OpenStreetMap API

const osmResponse = await fetch(`https://nominatim.openstreetmap.org/search?format=json&q=${validatedAddress}`);

if (!osmResponse.ok) {
    throw new Error('Failed to fetch latitude and longitude.');
}

const osmData = await osmResponse.json();

const { lat, lon } = osmData[0];

// Define the custom icon using L.icon

const customIcon = L.icon({  
    iconUrl: 'file:///Users/meghajotangya/Desktop/Green%20Way-Locate%20Nearby%20Landfills%20&%20Transfer%20Stations/location-pin-user-dot-icon.svg', // Replace 'path_to_your_custom_icon.svg' with the actual path or URL to your SVG icon  
    iconSize: [35, 35], // Adjust the size of the icon as needed  
    iconAnchor: [16, 32] // Adjust the anchor point if necessary  
});  
  
if (map) {
```

```

        map.remove();

    }

    map = L.map('mapContainer').setView([33.5987844812527,-112.077087523346],
13);

    L.tileLayer('https://s.tile.openstreetmap.org/{z}/{x}/{y}.png', {attribution:
'&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a>
contributors'}).addTo(map);

    marker = L.marker([lat, lon], { icon: customIcon }).addTo(map);

    marker.bindTooltip('<strong>Your Address:<br></strong>' + rowData[addressIndex]).openTooltip();

    break;

}

}

if (!isValid) {

    const validationResult = document.getElementById('validationResult');

    validationResult.style.display = 'block';

    validationResult.innerText = 'We could not find this address.';

    validationResult.style.color = 'red';

}

} catch (error) {

    console.error('Error:', error);

}

}

async function showNearestStation() {

    const addressInput =
        document.getElementById('addressInput').value.trim().toLowerCase(); // Convert input to
lowercase for case-insensitive comparison

```

```
try {  
    // Fetch latitude and longitude using OpenStreetMap API  
    const osmResponse = await fetch(`https://nominatim.openstreetmap.org/search?format=json&q=${addressInput}`);  
    if (!osmResponse.ok) {  
        throw new Error('Failed to fetch latitude and longitude.');//  
    }  
    const osmData = await osmResponse.json();  
    const { lat, lon } = osmData[0];  
  
    // Calculate distance to each station using Mapbox API  
    const distances = await Promise.all(stations.map(async (station) => {  
        const mapboxResponse = await fetch(`https://api.mapbox.com/directions/v5/mapbox/driving/${lon},${lat};${station.lon},${station.lat}?access_token=#####`);  
        if (!mapboxResponse.ok) {  
            throw new Error('Failed to calculate distance.');//  
        }  
        const mapboxData = await mapboxResponse.json();  
        return mapboxData.routes[0].distance;  
    }));  
  
    // Find the station with the shortest distance  
    const nearestIndex = distances.indexOf(Math.min(...distances));  
    const nearestStation = stations[nearestIndex];  
  
    // Create new map only if it's not already initialized  
    if (!map) {  
        map = L.map('mapContainer').setView([33.5987844812527,-112.077087523346], 13);  
    }  
}
```

```

L.tileLayer('https://tile.openstreetmap.org/{z}/{x}/{y}.png', {
    attribution: '&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors',
}).addTo(map);

}

// Display the nearest station on the map

if (nearestStation) {

    // Fetch route geometry using Mapbox Directions API and display on the map

    const routeResponse = await fetch(`https://api.mapbox.com/directions/v5/mapbox/driving/${nearestStation.lon},${nearestStation.lat};${nearestStation.id}, ${nearestStation.lat}?geometries=geojson&steps=true&access_token=#####`);

    const routeData = await routeResponse.json();

    const route = routeData.routes[0].geometry;

    // Add the route to the map

    L.geoJSON(route).addTo(map);

    // Add marker for the nearest station

    const stationMarker = L.marker([nearestStation.lat, nearestStation.lon]).addTo(map);

    stationMarker.bindTooltip(`<strong>Name:</strong> ${nearestStation.name} <br> <strong>Station Address:</strong> ${nearestStation.address}`).openTooltip();

    // Convert distance from meters to miles

    const distanceInMiles = distances[nearestIndex] * 0.000621371;

    // Update transfer station details on the web page with distance in miles

    document.getElementById('transferStationDetails').style.display = 'block';

    document.getElementById('stationName').innerHTML = `

        <span>${nearestStation.name}</span>

        <span class="station-distance">${distanceInMiles.toFixed(2)} mi</span>
    
```

```

';

document.getElementById('stationAddress').innerHTML = nearestStation.address ;

document.getElementById('stationHours').innerHTML = 'Monday-Friday: ' +
nearestStation.hours.mondayToFriday + '<br>' +
'Saturday: ' + nearestStation.hours.saturday + '<br>' +
'Sunday: ' + nearestStation.hours.sunday;

document.getElementById('stationHolidays').innerHTML =
nearestStation.holidays.join('<br>');

document.getElementById('stationDistance').innerHTML = "";

stationMarker.setIcon(L.icon({
    iconUrl: 'file:///Users/meghajotangiya/Desktop/Green%20Way-
Locate%20Nearby%20Landfills%20&%20Transfer%20Stations/location-pin-trash.svg', // 
Provide the path to your custom marker image
    iconSize: [35, 35],
    iconAnchor: [16, 32]
})); 

// Fetch direction using Mapbox Directions API and display it

const directionResponse = await
fetch(`https://api.mapbox.com/directions/v5/mapbox/driving/${lon},${lat};${nearestStation.l
on},${nearestStation.lat}?steps=true&access_token=#####`);

const directionData = await directionResponse.json();

const steps = directionData.routes[0].legs[0].steps;
const directions = steps.map(step => step.maneuver.instruction);

// Update transfer station details on the web page with directions

document.getElementById('stationDirection').innerHTML = directions.join('<br>');

}

} catch (error) {
    console.error('Error:', error);
}

```

```
}
```

```
// Function to update map container height based on white box height
```

```
function updateMapContainerHeight() {
```

```
    const whiteBox = document.getElementById('whiteBox');
```

```
    const whiteBoxHeight = whiteBox.offsetHeight;
```

```
    const transferDetailsBox = document.getElementById('transferStationDetails');
```

```
    const transferDetailsHeight = transferDetailsBox.offsetHeight;
```

```
    const webpageHeight = window.innerHeight;
```

```
    const mapContainer = document.getElementById('mapContainer');
```



```
// Calculate total height of white box including transfer station details
```

```
    const totalWhiteBoxHeight = whiteBoxHeight + transferDetailsHeight;
```



```
// Check if total white box height exceeds webpage height
```

```
    if (totalWhiteBoxHeight > webpageHeight) {
```

```
        // Set the map container's height to match the total white box height
```

```
        mapContainer.style.height = `${totalWhiteBoxHeight}px`;
```

```
    } else {
```

```
        // Set default height for the map container
```

```
        mapContainer.style.height = 'auto';
```

```
    }
```

```
}
```



```
// Call the updateMapContainerHeight function when the window is resized or when the
```

```
white box content changes
```

```
window.addEventListener('resize', updateMapContainerHeight);
```

```
document.addEventListener('DOMContentLoaded', updateMapContainerHeight); // Call
```

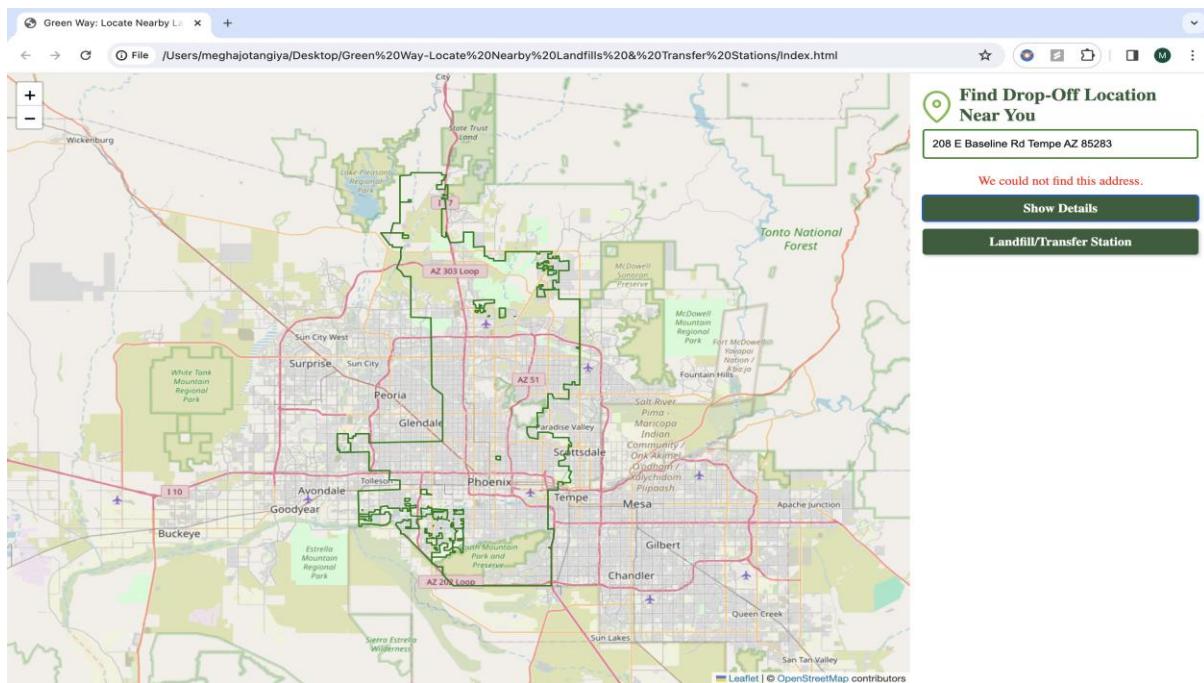
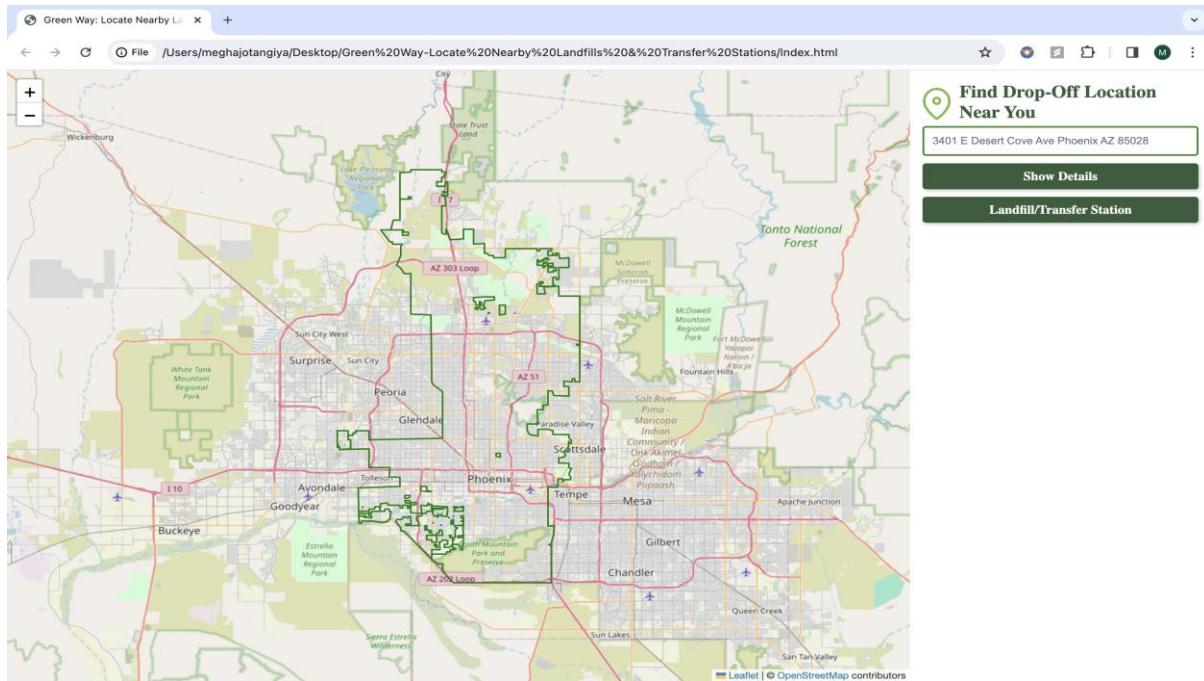
```
when the DOM content is loaded
```

```
</script>
```

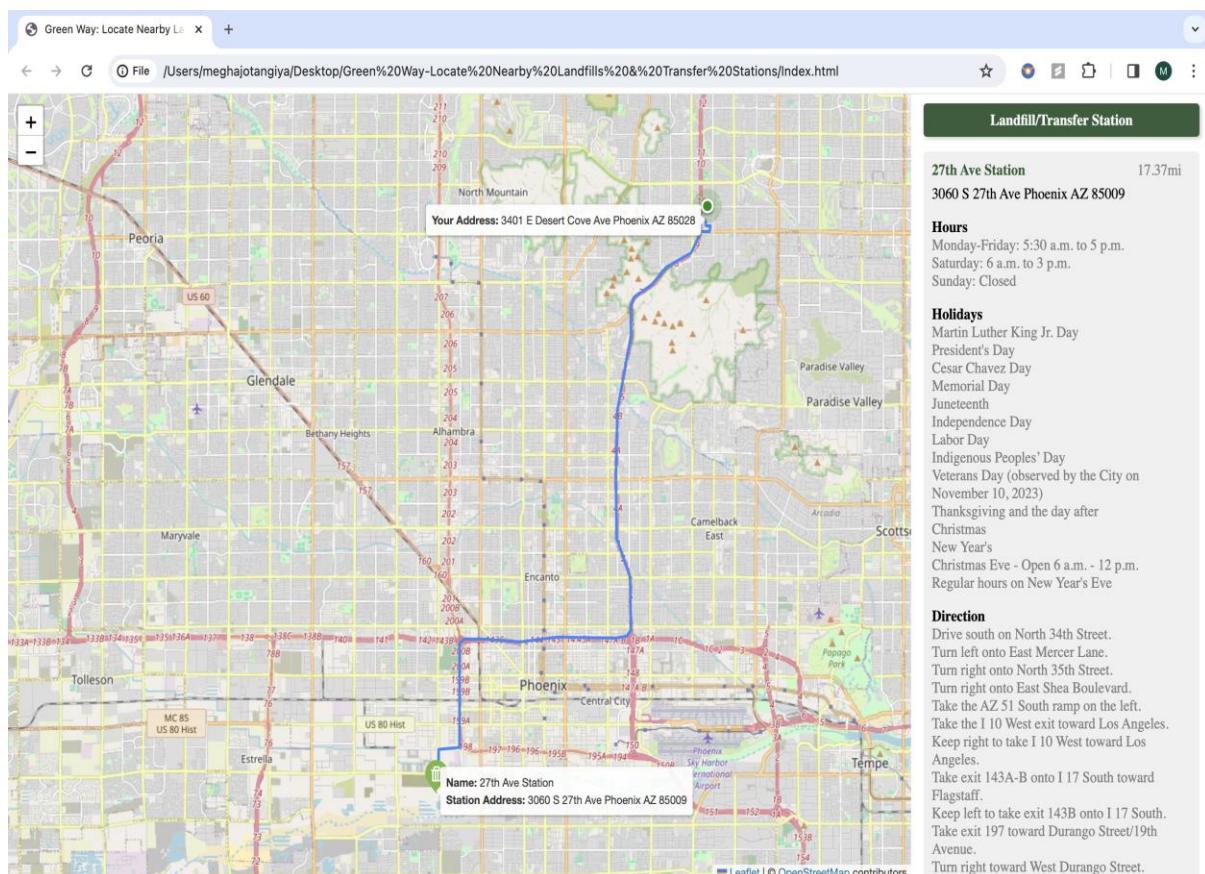
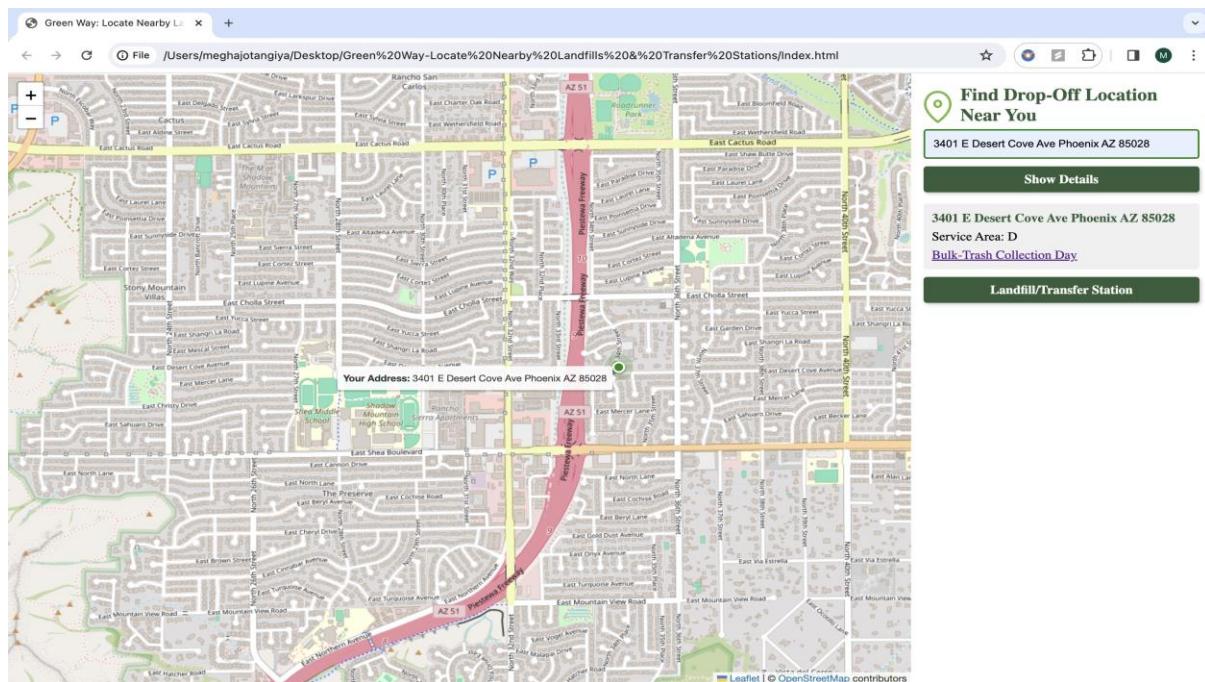
```
</body>
```

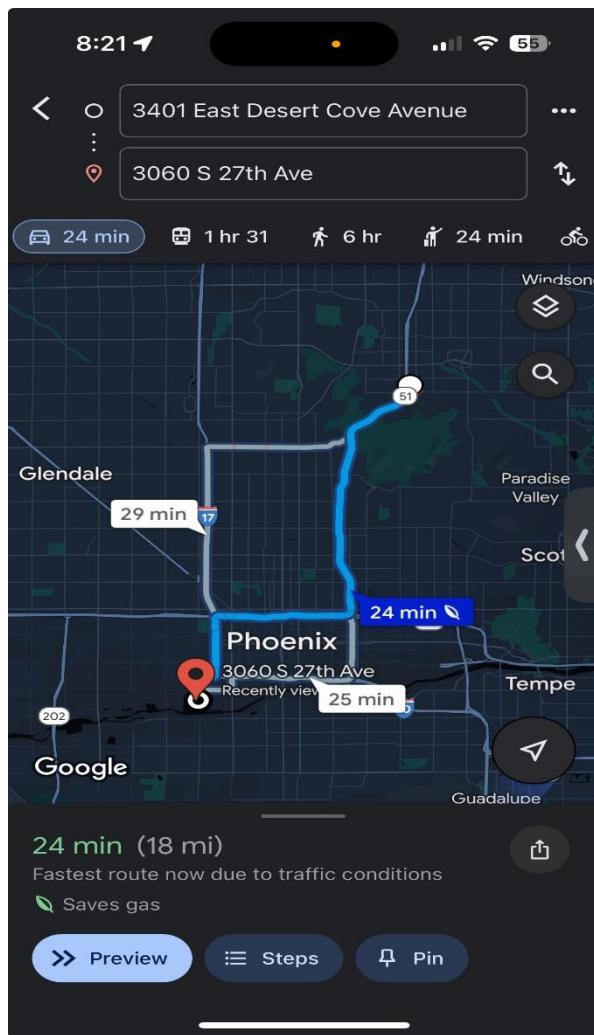
```
</html>
```

Screenshots:

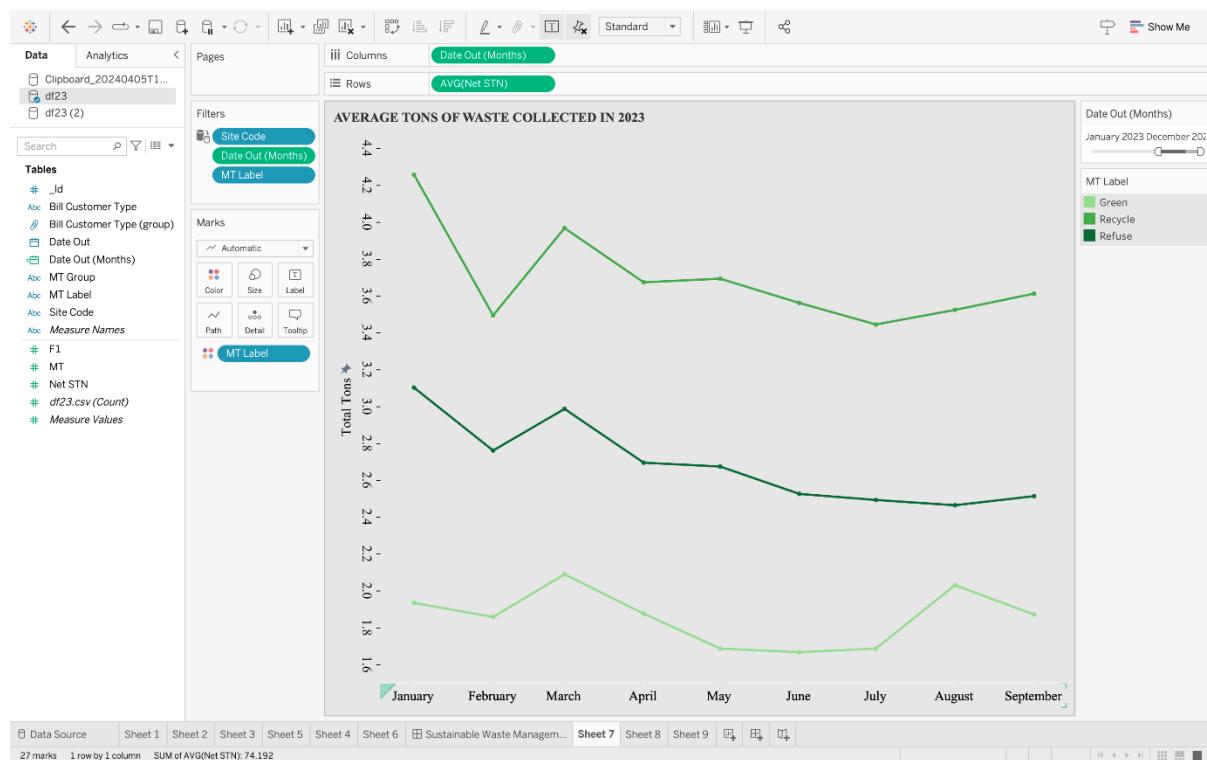


IFT 593 Final Project

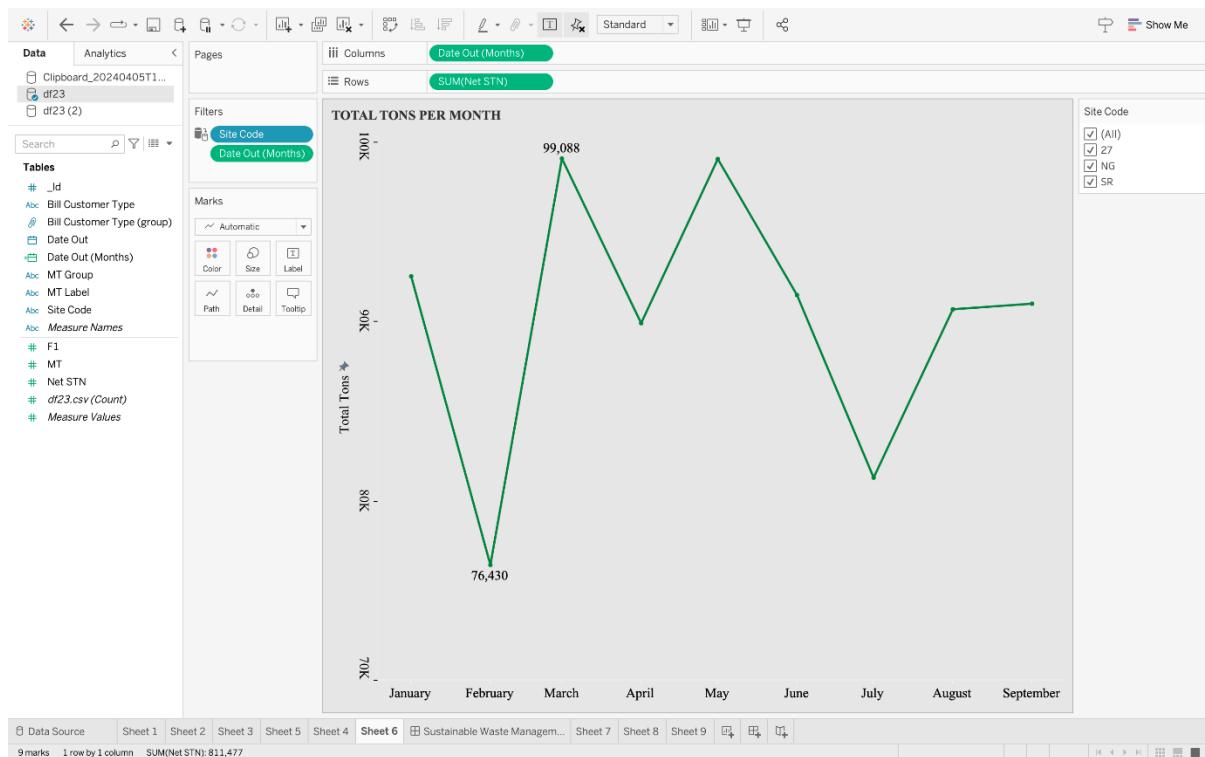




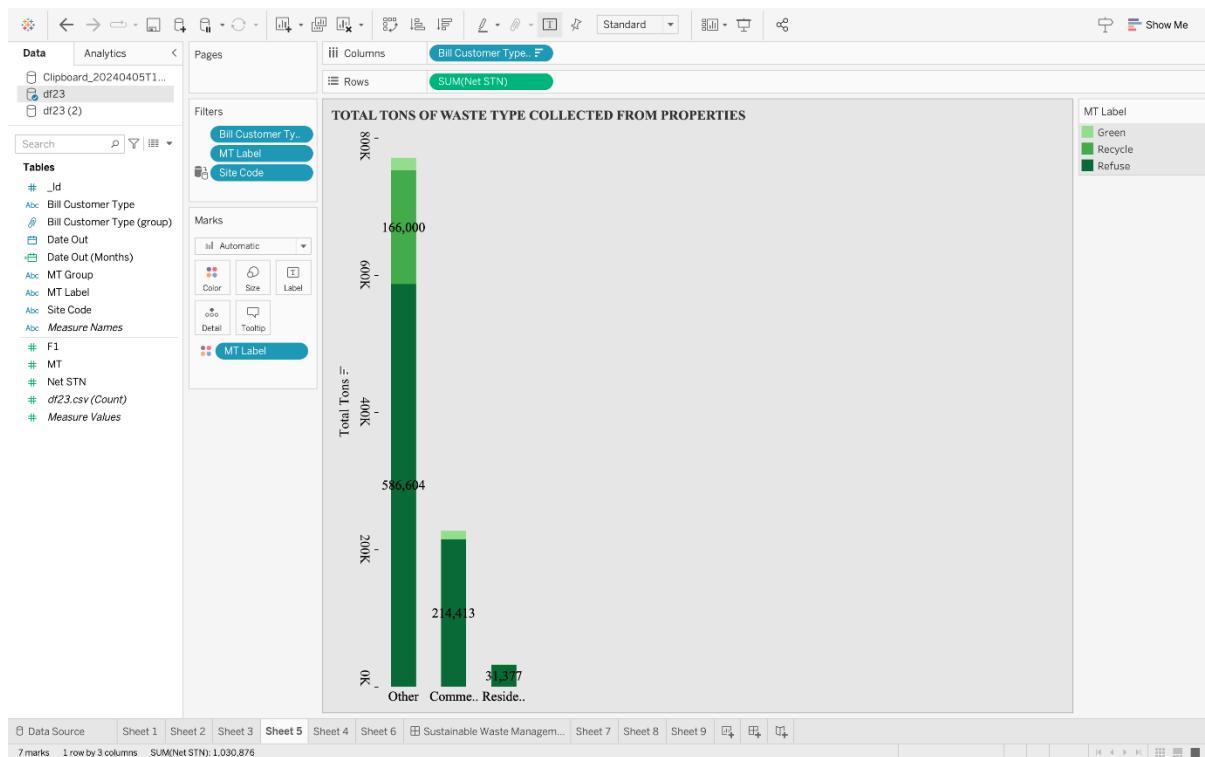
The Google Maps screenshot & the webpage, both display the distance from one location to the other. As we can see, the accuracy of both of these functions is almost similar with same routes. Also, the website displays public holidays on which the station might be closed.

Tableau Visualization:

The presented line chart offers a comprehensive analysis of the average monthly waste collection volumes categorized by waste type - Recycle, Refuse, and Green - within Phoenix City. Through a meticulous aggregation and categorization of collection data, the visualization provides valuable insights into the distribution and trends of waste collection across different months. By segmenting the data according to waste type, it enables a nuanced understanding of the relative contributions of Recycle, Refuse, and Green waste to the overall waste stream on a monthly basis. This detailed analysis facilitates the identification of patterns, anomalies, and potential areas for improvement in waste management practices. By visualizing the average tons of waste collected per month, categorized by waste type. This comprehensive portrayal of waste collection dynamics serves as a valuable resource for urban planners, waste management authorities, and policymakers, aiding in informed decision-making and strategic planning aimed at fostering a more efficient and environmentally sustainable waste management infrastructure within the city

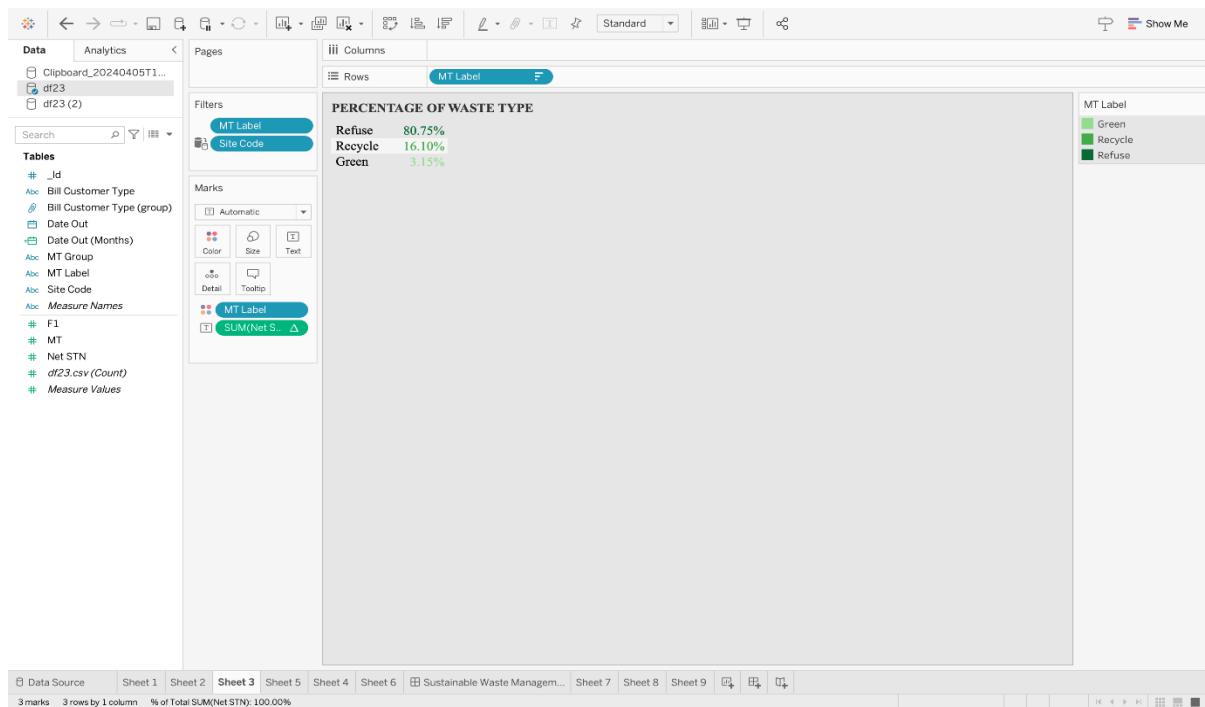


The line chart provides a temporal overview of monthly waste collection trends in Phoenix City, showcasing fluctuations in tonnage across different months. March emerges as the peak month, with a total collection of 99,000 tons, reflecting potentially heightened seasonal waste generation or specific events impacting waste volumes. Conversely, February records the lowest collection at 76,000 tons, suggesting potential factors such as reduced consumption or waste generation during this period. Analyzing these variations can inform waste management strategies, resource allocation, and operational planning to optimize collection efficiency and ensure timely waste disposal services. Understanding monthly waste generation patterns aids in maintaining effective waste management practices and promoting environmental sustainability in Phoenix City.

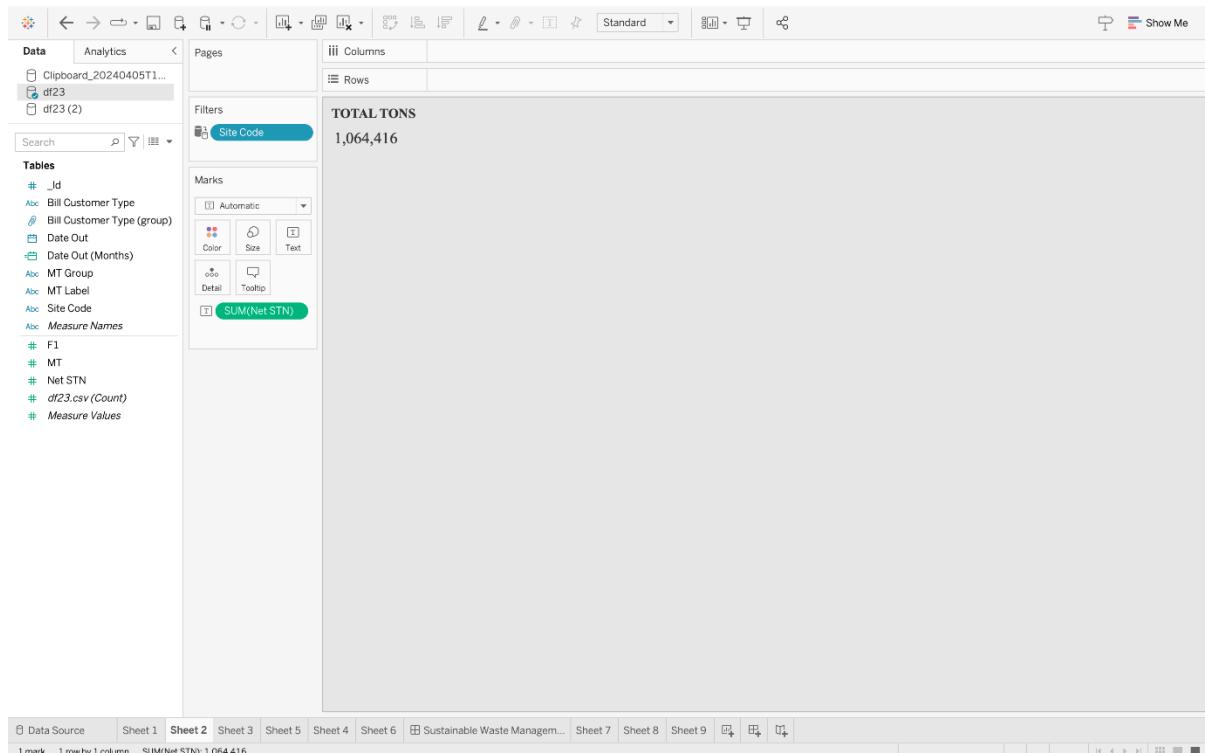


The categorical bar chart underscores the waste management dynamics across various property types in Phoenix City, with refuse waste dominating each category. This signifies the substantial volume of non-recyclable materials generated across residential, commercial, and industrial sectors. Addressing the management of refuse waste remains a critical challenge, necessitating comprehensive strategies for waste diversion and resource recovery. While specific tonnages for recycle and green waste are not delineated, their inclusion underscores the city's multifaceted approach to sustainable waste management. Promoting recycling initiatives and implementing green waste diversion programs are essential steps towards mitigating environmental impacts and fostering a circular economy ethos within Phoenix City.

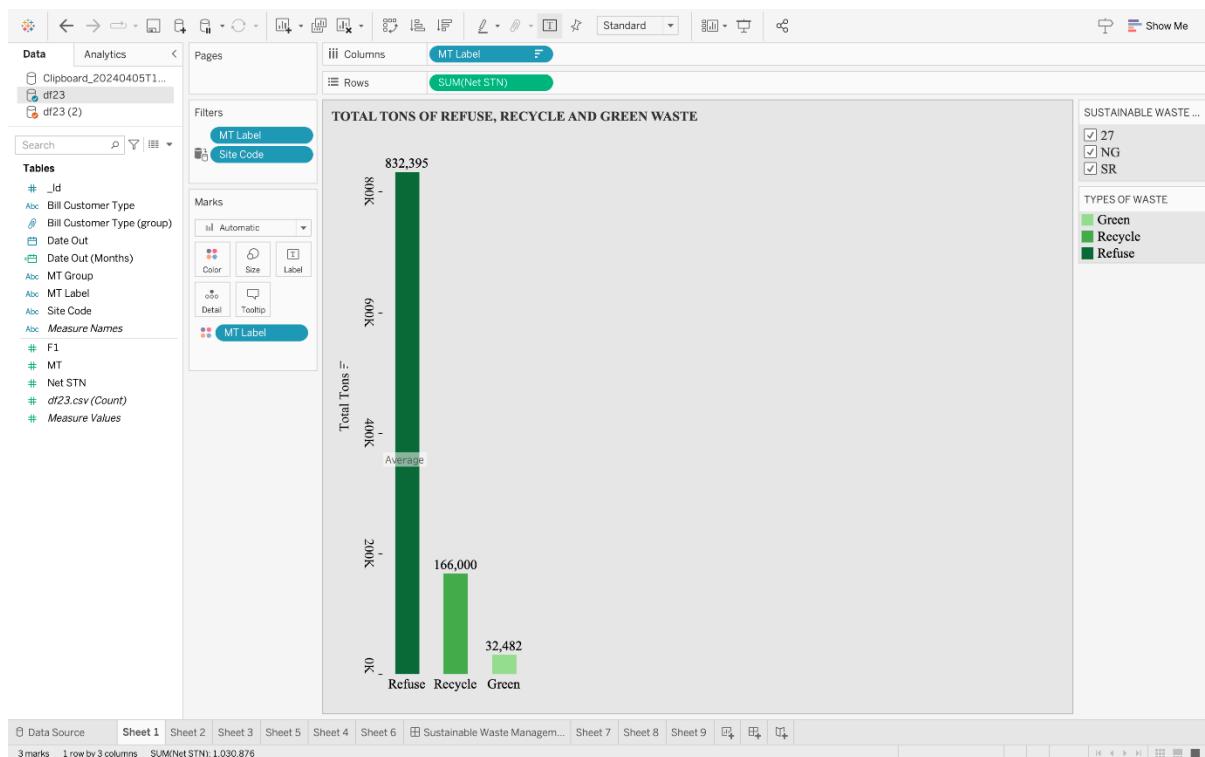
IFT 593 Final Project



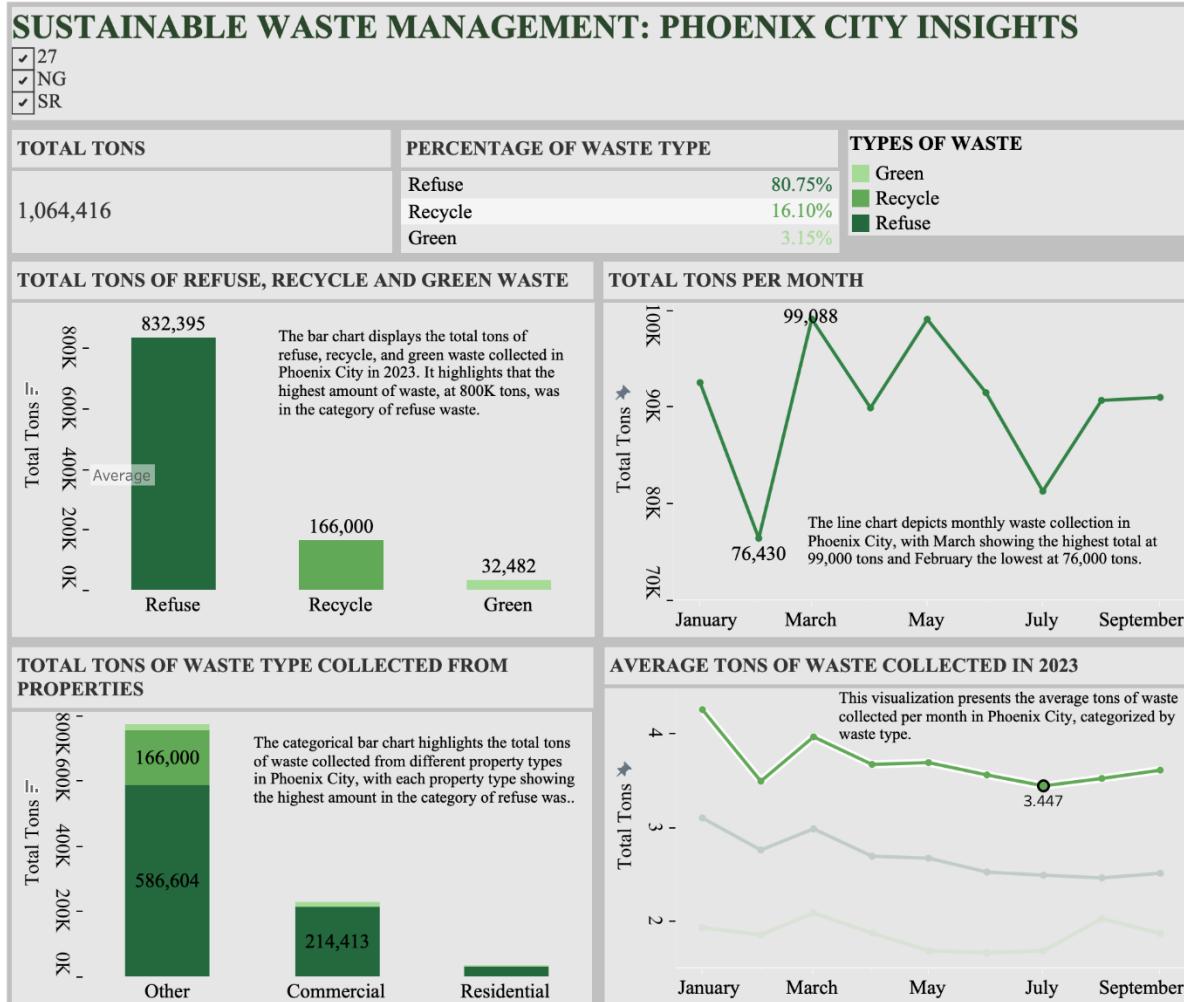
This graph displays the Percentage of Waste Type with Refuse being the highest type of waste collected with 80.75%, followed by Recycle (16.10) and Green (3.15%)



The total tons of waste collected till 2023 is 1,064,416.



The bar chart presents a comprehensive overview of waste management in Phoenix City for the year 2023, showcasing the total tons of refuse, recycle, and green waste collected. Notably, the data underscores the predominant volume of refuse waste, towering at an impressive 800,000 tons, marking it as the most substantial category among the three. This visualization provides crucial insights into the distribution of waste types, crucial for informed decision-making and sustainable urban planning.



As the culmination of our project, we are pleased to present our comprehensive dashboard, offering a detailed analysis of waste management dynamics in Phoenix City for the year 2023. Our dashboard encompasses a series of visualizations meticulously crafted to provide stakeholders with actionable insights into the city's waste landscape.

The first visualization lays the groundwork by presenting the total tons of waste collected, providing a foundational understanding of the scale of waste management efforts throughout the year.

In the second visualization, we delve deeper into the composition of waste types - Refuse, Recycle, and Green - presenting their respective percentages of contribution to the overall waste stream. This comparative analysis sheds light on the relative significance of each waste category in the broader context of waste management.

Our third visualization, a bar chart, highlights the total tons of refuse, recycle, and green waste collected, with a striking emphasis on the dominance of refuse waste, totaling 800,000 tons. This finding underscores the critical importance of targeted interventions to manage non-recyclable waste effectively.

Moving forward, our fifth visualization, a categorical bar chart, unveils the distribution of waste across different property types, revealing a consistent trend of refuse waste being the most prevalent category across diverse sectors.

The sixth and seventh visualizations, represented as line charts, offer insights into temporal trends in waste collection. While the former highlights monthly fluctuations in collection volumes, the latter presents the average tons of waste collected per month, categorized by waste type.

Together, these visualizations form a comprehensive narrative, empowering with the data-driven insights necessary for informed decision-making and strategic planning in the realm of sustainable waste management. We believe that this project contributes significantly to the ongoing discourse on environmental sustainability and underscores the imperative for concerted action in addressing contemporary waste challenges.

Insights & Actions:

Insights:

1. Operational Efficiency:

By analyzing the total tons collected at landfills and transfer stations in 2023, you can assess the operational efficiency of these waste processing facilities. Identifying peak periods of waste collection can help in resource allocation and scheduling.

2. Waste Composition:

The percentage table showing the breakdown of waste types (refuse, recycle, and green waste) gives insights into the city's waste composition. This information is valuable for designing recycling programs and promoting green initiatives.

3. Waste Management Trends:

The bar chart depicting the total tons of refuse, recycle, and green waste collected provides a comparative view of waste management trends. Identifying trends can aid in evaluating the effectiveness of waste reduction strategies.

4. Monthly Variations:

The line chart displaying total tons per month allows you to identify seasonal variations in waste generation and collection. This insight is useful for planning and adjusting waste management operations accordingly.

5. Residential Contribution:

Analyzing the total tons of waste type collected from properties helps in understanding the contribution of residential areas to overall waste generation. It can guide targeted outreach programs to encourage responsible waste disposal practices among residents.

6. Monthly Average:

Calculating the average tons of waste collected each month provides a benchmark for assessing the consistency of waste management efforts. Deviations from the average can highlight areas that require attention or improvement.

Actions:

1. Operational Efficiency Improvement:

- Implement optimized scheduling during peak waste collection periods at landfills and transfer stations to improve operational efficiency.
- Invest in technology and infrastructure upgrades to streamline waste processing and reduce bottlenecks.

2. Waste Composition Management:

- Develop targeted recycling programs based on the waste composition data to increase recycling rates and reduce landfill waste.
- Promote green initiatives such as composting for organic waste to divert it from landfills and encourage sustainable waste disposal practices.

3. Effective Waste Reduction Strategies:

- Analyze waste management trends to identify areas where waste reduction strategies can be most effective, such as reducing refuse or increasing recycling of specific materials.
- Collaborate with local businesses and industries to implement waste minimization practices and reduce waste generation at the source.

4. Seasonal Variation Planning:

- Use insights from seasonal waste variations to optimize waste collection routes, adjust staffing levels, and allocate resources efficiently during peak periods.

5. Community Engagement and Education:

- Launch targeted outreach programs based on residential waste contributions to educate residents about proper waste sorting, recycling techniques, and the importance of waste reduction.
- Organize community clean-up events, recycling drives, and educational workshops to foster a culture of sustainability and environmental responsibility.

6. Continuous Monitoring and Improvement:

- Regularly monitor and evaluate waste management efforts using metrics such as monthly average waste collection to identify areas for improvement.
- Implement feedback mechanisms and gather input from stakeholders to adapt strategies, address challenges, and achieve continuous improvement in waste management practices.

Tableau Public Link:

<https://public.tableau.com/app/profile/harshita.mishra3743/viz/SustainableWasteManagement-PhoenixCityInsights/SustainableWasteManagementPhoenixCityInsights?publish=yes>