Tanay Kulkarni

Andew ID: tskulkar

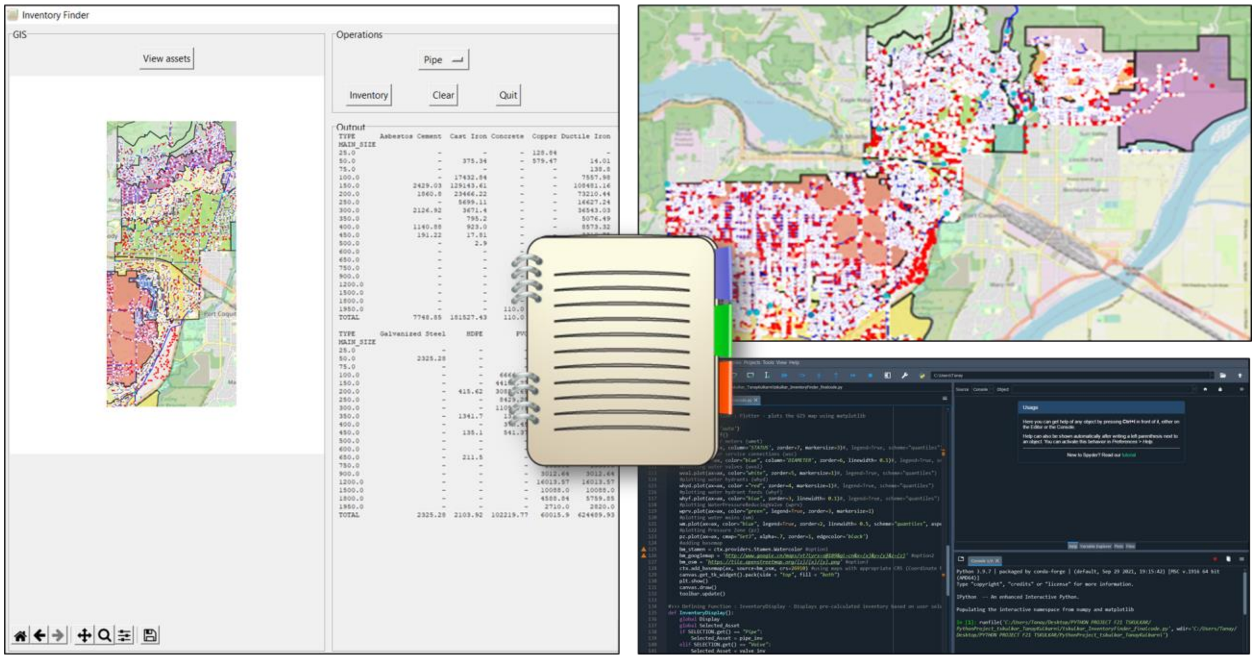
Mentor: Prof. Dr. Susan Finger

INVENTORY FINDER TOOL

12-746 FALL 2021

Department of Civil and Environmental Engineering

Carngie Mellon University



PROJECT REPORT

OCTOBER 15, 2021

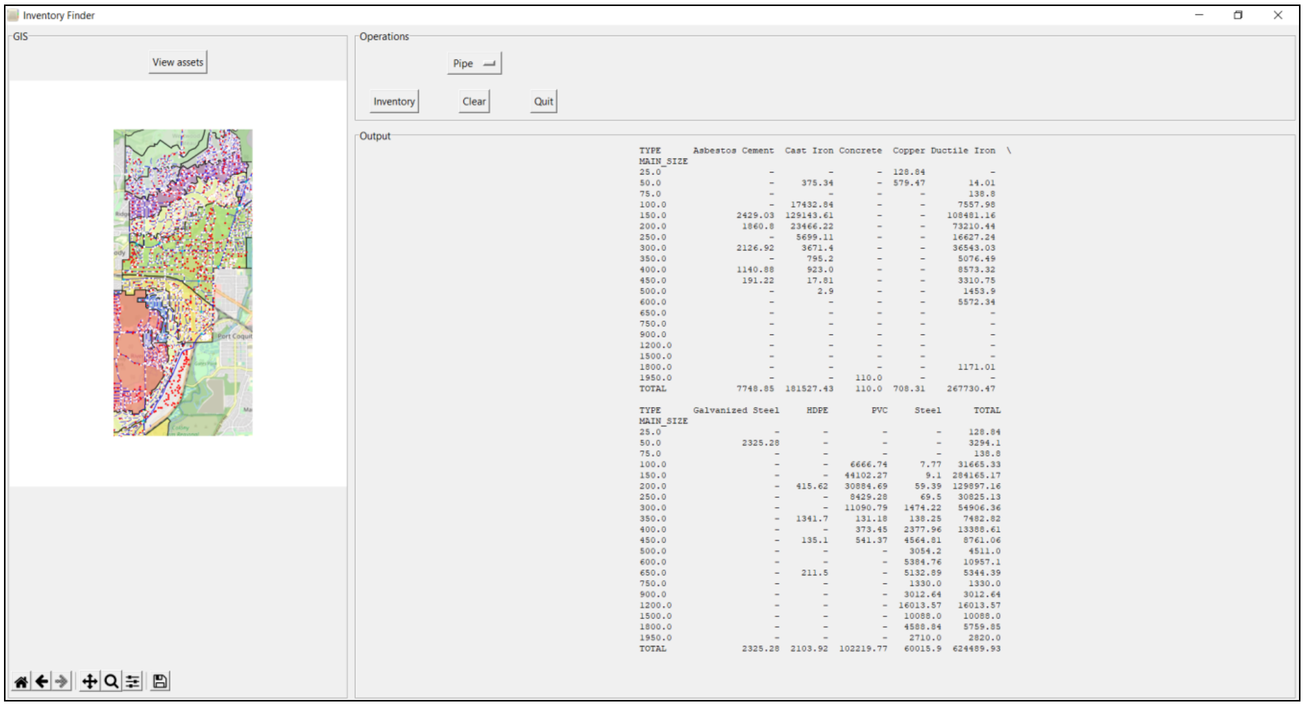
Contents

Table of Figures

1. Introduction

The municipal water and wastewater network infrastructure lie underneath the city along its roads, and this network infrastructure needs periodic maintenance, check-ups, and repairs to ensure sustainability and performance. The construction and maintenance engineers need to understand the actual asset inventory in a particular region. For example, one may be interested in understanding the diameter-wise count of pressure-reducing valves in a particular subnetwork. Understanding the asset inventory on the field is complex, and this program attempts to help engineers retrieve a quick ‘ready to use’ inventory of the elements of a Water Distribution Network (WDN). The program has a straightforward Graphical User Interface (GUI), and the user requires no prior knowledge of the water network hydraulics.

2. Background

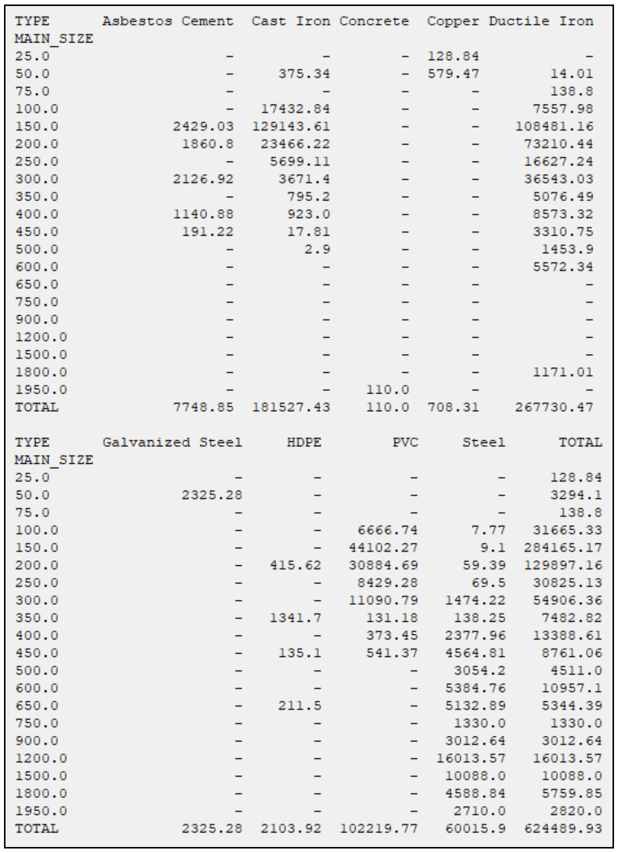


```
TYPE       Asbestos Cement  Cast Iron Concrete  Copper Ductile Iron  \
MAIN_SIZE
25.0                     -          -        -  128.84            -
50.0                     -     375.34        -  579.47        14.01
75.0                     -          -        -       -        138.8
100.0                    -   17432.84        -       -      7557.98
150.0              2429.03  129143.61        -       -    108481.16
200.0               1860.8   23466.22        -       -     73210.44
250.0                    -    5699.11        -       -     16627.24
300.0              2126.92     3671.4        -       -     36543.03
350.0                    -      795.2        -       -      5076.49
400.0              1140.88      923.0        -       -      8573.32
450.0               191.22      17.81        -       -      3310.75
500.0                    -        2.9        -       -       1453.9
600.0                    -          -        -       -      5572.34
650.0                    -          -        -       -            -
750.0                    -          -        -       -            -
900.0                    -          -        -       -            -
1200.0                   -          -        -       -            -
1500.0                   -          -        -       -            -
1800.0                   -          -        -       -      1171.01
1950.0                   -          -    110.0       -            -
TOTAL              7748.85  181527.43    110.0  708.31    267730.47

TYPE       Galvanized Steel     HDPE        PVC     Steel      TOTAL
MAIN_SIZE
25.0                      -        -          -         -     128.84
50.0                2325.28        -          -         -     3294.1
75.0                      -        -          -         -      138.8
100.0                     -        -    6666.74      7.77   31665.33
150.0                     -        -   44102.27       9.1  284165.17
200.0                     -   415.62   30884.69     59.39  129897.16
250.0                     -        -    8429.28      69.5   30825.13
300.0                     -        -   11090.79   1474.22   54906.36
350.0                     -   1341.7     131.18    138.25    7482.82
400.0                     -        -     373.45   2377.96   13388.61
450.0                     -    135.1     541.37   4564.81    8761.06
500.0                     -        -          -    3054.2     4511.0
600.0                     -        -          -   5384.76    10957.1
650.0                     -    211.5          -   5132.89    5344.39
750.0                     -        -          -    1330.0     1330.0
900.0                     -        -          -   3012.64    3012.64
1200.0                    -        -          -  16013.57   16013.57
1500.0                    -        -          -   10088.0    10088.0
1800.0                    -        -          -   4588.84    5759.85
1950.0                    -        -          -    2710.0     2820.0
TOTAL               2325.28  2103.92  102219.77   60015.9  624489.93
```

Figure 1 The Program GUI

The program offers a simple GUI (see Fig.1) to view the water distribution network assets on GIS with an aerial background map. The GUI is divided into three frames viz. ‘GIS,’ ‘Operations,’ and ‘Output.’ The ‘GIS’ frame offers a dynamic display of the network assets by accessing the shapefiles data with aerial imagery. The ‘Operations’ frame offers various essential operations such as selecting the asset, generating the inventory output, clearing any previously generated inventory, and closing the program. The ‘Operations’ frame has a ‘drop-down list’ to select the element of interest out of total assets that the user might have added (e.g., pipes), an ‘Inventory’ button, a ‘Clear’ button, and a ‘Quit’ button. The drop-down list allows the user to select one specific element of interest to create the inventory. The ‘Inventory’ button generates and displays the inventory in the ‘Output’ frame. In the backend, the program also generates an excel output for the inventories of all the elements. A typical inventory that the program generates is shown below in Fig. 2.



```
TYPE      Asbestos Cement  Cast Iron Concrete  Copper Ductile Iron
MAIN_SIZE
25.0                    -          -        -  128.84            -
50.0                    -     375.34        -  579.47        14.01
75.0                    -          -        -       -        138.8
100.0                   -   17432.84        -       -      7557.98
150.0             2429.03  129143.61        -       -    108481.16
200.0              1860.8   23466.22        -       -     73210.44
250.0                   -    5699.11        -       -     16627.24
300.0             2126.92     3671.4        -       -     36543.03
350.0                   -      795.2        -       -      5076.49
400.0             1140.88      923.0        -       -      8573.32
450.0              191.22      17.81        -       -      3310.75
500.0                   -        2.9        -       -       1453.9
600.0                   -          -        -       -      5572.34
650.0                   -          -        -       -            -
750.0                   -          -        -       -            -
900.0                   -          -        -       -            -
1200.0                  -          -        -       -            -
1500.0                  -          -        -       -            -
1800.0                  -          -        -       -      1171.01
1950.0                  -          -    110.0       -            -
TOTAL             7748.85  181527.43    110.0  708.31    267730.47

TYPE      Galvanized Steel     HDPE         PVC     Steel       TOTAL
MAIN_SIZE
25.0                     -        -           -         -      128.84
50.0               2325.28        -           -         -      3294.1
75.0                     -        -           -         -       138.8
100.0                    -        -     6666.74      7.77    31665.33
150.0                    -        -    44102.27       9.1   284165.17
200.0                    -   415.62    30884.69     59.39   129897.16
250.0                    -        -     8429.28      69.5    30825.13
300.0                    -        -    11090.79   1474.22    54906.36
350.0                    -   1341.7      131.18    138.25     7482.82
400.0                    -        -      373.45   2377.96    13388.61
450.0                    -    135.1      541.37   4564.81     8761.06
500.0                    -        -           -    3054.2      4511.0
600.0                    -        -           -   5384.76     10957.1
650.0                    -    211.5           -   5132.89     5344.39
750.0                    -        -           -    1330.0      1330.0
900.0                    -        -           -   3012.64     3012.64
1200.0                   -        -           -  16013.57    16013.57
1500.0                   -        -           -   10088.0     10088.0
1800.0                   -        -           -   4588.84     5759.85
1950.0                   -        -           -    2710.0      2820.0
TOTAL              2325.28  2103.92   102219.77   60015.9   624489.93
```

Figure 2 Pipe Inventory

3. Implementation

3.1 Data requirements

Following are the data requirements for the project:

3.1.1 Input Data Requirements

- The GIS shapefiles of various elements of water distribution networks, viz. pipes, tanks, pumps, valves, and hydrants, are essential.
- The meta-data, i.e., attribute data of individual elements, to generate an inventory.

3.1.2 Input Data Source

Water Network Infrastructure Dataset (valves, water mains, service connections, meters, pressure reducing valves, hydrants, reservoirs, pumps) for Coquitlam, Canada is available. The shapefiles are available [1] at the ESRI website, and the data is royalty-free [2].

3.1.3 Software Requirements

The program was coded in Python (Version 3.9.7) using Anaconda Spyder (Version 5.0.5) on a Windows 10 Operating System with 16GB RAM and Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz processor.

3.2 The Methodology and Code Chunks

The overall flow of the program code is given in Fig. 3. The entire code can be found in Appendix 1 of this report.

The chunks of codes below briefly illustrates the use of different methods within the main code.

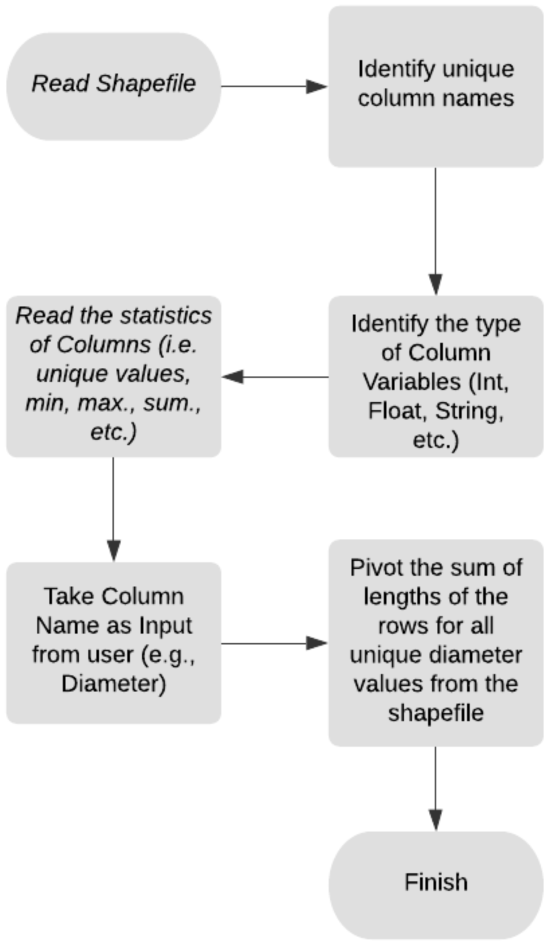

Read Shapefile
Identify unique column names
Identify the type of Column Variables (Int, Float, String, etc.)
Read the statistics of Columns (i.e. unique values, min, max., sum., etc.)
Take Column Name as Input from user (e.g., Diameter)
Pivot the sum of lengths of the rows for all unique diameter values from the shapefile
Finish

Figure 3 Methodology workflow

Importing Modules and Packages:

The following modules and packages are installed and used in the code:

```
import pandas as pd

import geopandas as gpd

import matplotlib.pyplot as plt

import contextily as ctx

import tkinter as tk

from matplotlib.figure import Figure

from matplotlib.backends.backend_tkagg import (FigureCanvasTkAgg, NavigationToolbar2Tk)

pd.set_option("display.max_columns", None)
```

Creating the GUI window:

Tkinter is used as a method to create the GUI.

```
root = tk.Tk()

root.iconbitmap('ProjectICON.ico')

root.title("Inventory Finder")

width= root.winfo_screenwidth()

height= root.winfo_screenheight()

root.geometry("%dx%d" % (width, height))
```

Creating the frames within GUI window:

Three Label Frames created within the GUI viz., ‘GIS,’ ‘Operations,’ and ‘Output’.

```
frame1 = tk.LabelFrame(root, text="GIS", width=250, height=850)

frame1.pack(side = "left", fill="y", pady=5, padx=5, expand=False)
```

Reading the shapefiles:

Shapefiles for meters, service connections, valves, hydrants, hydrant feeds, Pressure Reducing Valves (PRV), mains, and pressure zones are read using `pandas.read_file()` and assigned to variables.

```
wmet = gpd.read_file('Water_Meters.shp')
```

Creating inventory pivot tables

Pivot tables are generated for mains, valves, hydrants, and meters using `pandas.pivot_table()` as shown below:

```
pipe_inv = pd.pivot_table(wm, values = 'LENGTH', index='MAIN_SIZE', columns='TYPE', aggfunc='sum', fill_value='-', margins = 
True, margins_name="TOTAL")
```

Functions defined in the code:

The code consists of three functions, `plotter()`, `InventoryDisplay()`, and `clear()` for different operations. The `plotter()` plots the GIS shapefiles on a canvas within frame1.

```
def plotter():

    ax.set_aspect('auto')

    ax.set_axis_off()

    wmet.plot(ax=ax, column='STATUS', zorder=7, markersize=3)
    wsc.plot(ax=ax, color="blue", column='DIAMETER', zorder=6, linewidth= 0.1)
    wval.plot(ax=ax, color="white", zorder=5, markersize=1)
    whyd.plot(ax=ax, color ="red", zorder=4, markersize=1)
    whyf.plot(ax=ax, color="blue", zorder=3, linewidth= 0.1)
    wprv.plot(ax=ax, color="green", legend=True, zorder=3, markersize=1)
    wm.plot(ax=ax, color="blue", legend=True, zorder=2, linewidth= 0.5, scheme="quantiles", aspect="auto")
    pz.plot(ax=ax, cmap="Set3", alpha=.7, zorder=1, edgecolor='black')


    bm_osm = 'https://tile.openstreetmap.org/{z}/{x}/{y}.png'
    ctx.add_basemap(ax, source=bm_osm, crs=26910)
    canvas.get_tk_widget().pack(side = "top", fill = "both")

    plt.show()
    canvas.draw()
    toolbar.update()
```

A simple drop-down option is developed for the user to first select the desired element to generate the inventory and before calling the `InventoryDisplay()`.

```
options =["Pipe",
        "Valve",
        "Hydrant",
        "Meter"]
SELECTION = tk.StringVar()
Selected_Asset = ""
SELECTION.set("Select Asset")
drop = tk.OptionMenu(frame2, SELECTION, *options)
drop.grid(row=0, column=2, padx=20, pady=10)
```

The `InventoryDisplay()` displays the inventory within the frame3 and creates an Excel Output.xlxs in the program folder.

```
def InventoryDisplay():
    global Display
    global Selected_Asset
    if SELECTION.get() == "Pipe":
        Selected_Asset = pipe_inv
    elif SELECTION.get() == "Valve":
        Selected_Asset = valve_inv
    elif SELECTION.get() == "Hydrant":
        Selected_Asset = hydrant_inv
    elif SELECTION.get() == "Meter":
        Selected_Asset = meters_inv
    Display.destroy()
    Display =tk.Label(frame3, text= Selected_Asset ,font=("Courier", 8))
    with pd.ExcelWriter('output.xlsx') as writer:
        pipe_inv.to_excel(writer, sheet_name='PipeInventory')
        valve_inv.to_excel(writer, sheet_name='ValveInventory')
        hydrant_inv.to_excel(writer, sheet_name='HydrantInventory')
        meters_inv.to_excel(writer, sheet_name='MeterInventory')
    Display.pack()
```

The `clear()` function clears the frame3 when called.

```
def clear():
    Display.destroy()
    Display.pack()
```

Buttons:

The ‘View assets,’ ‘Inventory,’ ‘Clear,’ and ‘Quit’ Buttons were created to call the above described functions.

```
invent_button = tk.Button(frame2, text="Inventory", command=InventoryDisplay)
invent_button.grid(row=1, column=1, padx=20, pady=10)
```

4. Conclusions

4.1 Python: Learning Objectives – Hits and Misses

Individual learning objectives accomplished for this project are as follows:

- Writing a professional code in Python
- Learning to articulate the problem definition in-terms of programming logic and writing algorithms to solve a particular problem
- Understanding how different Python modules such as Pandas and GeoPandas are used to interact with GIS data within a Python interface

Individual learning objectives that could not be accomplished for this project are as follows:

- Exporting plots and inventory to different data formats such as PDF, JPEG, etc.
- Importing shapefiles through user-activity.
- Dynamic i.e. zoom-based or extent-based inventory generation.

These objectives could not be achieved because of time limitation.

4.2 Deliverables

The project delivers a robust GUI to read GIS shapefiles and display inventory for assets on the GUI as well as an excel spreadsheet.

4.3 Future Possibilities for this Project

This project has the potential to upscale to an enterprise-level Asset Management Platform. Beyond the desktop application, it has the potential for any GPS-enabled hand-held devices for field engineers.

5. References

1. https://www.arcgis.com/apps/mapviewer/index.html?layers=83ea9a808f6b455299997545a2e946ab, accessed on 9/21/2021
2. https://www.coquitlam.ca/894/Open-Government-Licence, accessed on 9/21/2021
3. https://www.python.org/ accessed on 9/21/2021
4. https://tkdocs.com/shipman/ accessed on 10/15/2021
5. https://greenteapress.com/wp/think-python-2e/ accessed on 10/13/2021
6. https://www.youtube.com/c/Codemycom accessed on 10/15/2021
7. https://geopandas.org/ accessed on 10/15/2021

6. Python Code

```
#Python Vesion 3.9.7

#IDE Spyder  5.0.5

#Date - October 2021

#SubjectCode - 12746

#SubjectName - Introduction to Python Prototyping of Infrastructure Systems

#Advisor - Prof. Dr. Susan Finger

#Student - Mr. Tanay Kulkarni

#ANDREW ID - TSKULKAR

#Department - CEE, Carnegie Mellon University


#>>> What does this program do?

#>>> This program reads a few GIS shapefiles in the backend and generates a pivot table that displays

#>>> the inventory of the assets such as Pipes, Valves, Meters, and Hydrants. In the frontend, on the GUI

#>>> the user can view these GIS assets mapped on a dynamic aerial imagery with zoom in/out functionality.

#>>> The use then can select a particular asset of interest(e.g., Pipes) from a drop-down menu to view the

#>>> inventory of those assets inside the GUI.


#MainReferences:

    #https://www.python.org/

    #https://tkdocs.com/shipman/

    #https://www.youtube.com/c/Codemycom

    #https://greenteapress.com/wp/think-python-2e/


#>>> Modules and Packages installed/ required

import pandas as pd

import geopandas as gpd

import matplotlib.pyplot as plt

import contextily as ctx

import tkinter as tk

from matplotlib.figure import Figure

from matplotlib.backends.backend_tkagg import (FigureCanvasTkAgg, NavigationToolbar2Tk)


pd.set_option("display.max_columns", None) #this ensures that Python doesn't hide column names
```

```
#>>> Creating a new GUI window
root = tk.Tk()
root.iconbitmap('ProjectICON.ico') #source: https://iconarchive.com/tag/inventory-ico
root.title("Inventory Finder")
width= root.winfo_screenwidth()
height= root.winfo_screenheight()
root.geometry("%dx%d" % (width, height))

#>>> Creating Frames for better visualization and display of the overall program
frame1 = tk.LabelFrame(root, text="GIS", width=250, height=850)
frame1.pack(side = "left", fill="y", pady=5, padx=5, expand=False)

frame2 = tk.LabelFrame(root, text="Operations", width=700, height=250)
frame2.pack(side = "top", fill="both", pady=5, padx=5, expand=False)

frame3 = tk.LabelFrame(root, text="Output", width=700, height=500)
frame3.pack(side = "bottom", fill="both", pady=5, padx=5, expand=True)

#>>> Reading the shapefiles
## Water Meters
wmet = gpd.read_file('Water_Meters.shp')
## Water Service Connections
wsc = gpd.read_file('Water_Service_Connections.shp')
## Water Valves
wval = gpd.read_file('Water_Valves.shp')
## Water Hydrants
whyd = gpd.read_file('Water_Hydrants.shp')
## Water Hydrant Feeds
whyf = gpd.read_file('Water_Hydrant_Feeds.shp')
## Water PRV
wprv = gpd.read_file('Water_PRV.shp')
## Water Mains
wm = gpd.read_file('Water_Mains.shp')
## Water Pressure Zones
```

```
pz = gpd.read_file('Water_Pressure_Zones.shp')


#>>> Creating a Canvas to display the Plotted Figure
fig, ax = plt.subplots(figsize=(5,6), dpi =100)
canvas = FigureCanvasTkAgg(fig, master = frame1)
toolbar = NavigationToolbar2Tk(canvas, frame1)


#>>> Creating ready-to-use 'inventory' variable by using pivot_table from pandas
#for pipes
pipe_inv = pd.pivot_table(wm, values = 'LENGTH', index='MAIN_SIZE', columns='TYPE', aggfunc='sum', fill_value='-', margins =
True, margins_name="TOTAL")
#for valves
valve_inv = pd.pivot_table(wval, values = 'VALVE_STS', index='VALVE_SIZE', columns='FUNCTION', aggfunc='count',
fill_value='-', margins = True, margins_name="TOTAL")
#for water hydrants
hydrant_inv = pd.pivot_table(whyd,  values = 'STATUS', index='MODEL', columns='FLOW_CLASS', aggfunc='count',
fill_value='-', margins = True, margins_name="TOTAL")
#for water meters
meters_inv = pd.pivot_table(wmet,  values = 'STATUS', index='DIAMETER', aggfunc='count', fill_value='-', margins = True,
margins_name="TOTAL")


#>>> Adding a dropdown for user to select the asset
options =["Pipe",
        "Valve",
        "Hydrant",
        "Meter"]
SELECTION = tk.StringVar()
Selected_Asset = ""
SELECTION.set("Select Asset")
drop = tk.OptionMenu(frame2, SELECTION, *options)
drop.grid(row=0, column=2, padx=20, pady=10)


global Display
Display =tk.Label(frame3,font=("Courier", 8))
Display.pack()


#>>> Defining Function : Plotter - plots the GIS map using matplotlib
```

```
def plotter():
    ax.set_aspect('auto')
    ax.set_axis_off()
    #plotting water meters (wmet)
    wmet.plot(ax=ax, column='STATUS', zorder=7, markersize=3)#, legend=True, scheme="quantiles")
    #plotting water service connections (wsc)
    wsc.plot(ax=ax, color="blue", column='DIAMETER', zorder=6, linewidth= 0.1)#, legend=True, scheme="quantiles")
    #plotting water valves (wval)
    wval.plot(ax=ax, color="white", zorder=5, markersize=1)#, legend=True, scheme="quantiles")
    #plotting water hydrants (whyd)
    whyd.plot(ax=ax, color ="red", zorder=4, markersize=1)#, legend=True, scheme="quantiles")
    #plotting water hydrant feeds (whyf)
    whyf.plot(ax=ax, color="blue", zorder=3, linewidth= 0.1)#, legend=True, scheme="quantiles")
    #plotting WaterPressureReducingValve (wprv)
    wprv.plot(ax=ax, color="green", legend=True, zorder=3, markersize=1)
    #plotting water mains (wm)
    wm.plot(ax=ax, color="blue", legend=True, zorder=2, linewidth= 0.5, scheme="quantiles", aspect="auto")
    #plotting Pressure Zone (pz)
    pz.plot(ax=ax, cmap="Set3", alpha=.7, zorder=1, edgecolor='black')
    #adding basemap
    bm_stamen = ctx.providers.Stamen.Watercolor #option1
    bm_googlemap = 'http://www.google.cn/maps/vt?lyrs=s@189&gl=cn&x={x}&y={y}&z={z}' #option2
    bm_osm = 'https://tile.openstreetmap.org/{z}/{x}/{y}.png' #option3
    ctx.add_basemap(ax, source=bm_osm, crs=26910) #using maps with appropriate CRS (Coordinate Reference System)
    canvas.get_tk_widget().pack(side = "top", fill = "both")
    plt.show()
    canvas.draw()
    toolbar.update()

#>>> Defining Function : InventoryDisplay - Displays pre-calculated inventory based on user selection
def InventoryDisplay():
    global Display
    global Selected_Asset
    if SELECTION.get() == "Pipe":
        Selected_Asset = pipe_inv
    elif SELECTION.get() == "Valve":
        Selected_Asset = valve_inv
    elif SELECTION.get() == "Hydrant":
        Selected_Asset = hydrant_inv
    elif SELECTION.get() == "Meter":
        Selected_Asset = meters_inv
    Display.destroy()
    Display =tk.Label(frame3, text= Selected_Asset ,font=("Courier", 8)) #important to note that selected font is monospaced! -
ensures better display of tables
    with pd.ExcelWriter('output.xlsx') as writer: #ExcelWriter of pandas is used to create a complimentary Excel File that
records inventory in the background
        pipe_inv.to_excel(writer, sheet_name='PipeInventory')
        valve_inv.to_excel(writer, sheet_name='ValveInventory')
        hydrant_inv.to_excel(writer, sheet_name='HydrantInventory')
        meters_inv.to_excel(writer, sheet_name='MeterInventory')
    Display.pack()

#>>> Defining Function : Clear - Clears the output display frame
def clear():
    Display.destroy()
    Display.pack()

#>>> Creating Buttons within various frames
# Buttons in Frame1
assets_button = tk.Button(frame1, text="View assets", command=plotter)
assets_button.pack(padx=20, pady=10)
# INVENTORY Button in Frame2
invent_button = tk.Button(frame2, text="Inventory", command=InventoryDisplay)
invent_button.grid(row=1, column=1, padx=20, pady=10)
# CLEAR Button in Frame2
clear_button = tk.Button(frame2, text="Clear", command=clear)
clear_button.grid(row=1, column=2, padx=20, pady=10)
# QUIT Button in Frame2
quit_button = tk.Button(frame2, text="Quit", command=root.destroy)
quit_button.grid(row=1, column=3, padx=20, pady=10)
```

```
#>>> Closing the loop for root window
root.mainloop()
```