Patrolling Emission in China

The Carbon Cops

Andrew Goetz, Nathan Evans, Emily Arnold, Evans Bowyer, Kaleb Sims

Emily Arnold - Scrum Master

- Graduated from Auburn High School, Auburn, Alabama.
- Pursuing an Aerospace Engineering degree from Auburn University.
- Also, pursuing a minor in Aviation Management.
- Desires to work for a Aeronautical Company post graduation.

Andrew Goetz - Product Owner

- Graduated from Sparkman High School, Huntsville, Alabama.
- Pursuing a Applied Mathematics degree from Auburn University.
- Also, pursuing a minor in Statistics.
- Desires a secondary degree and data science career post graduation.

Nathan Evans - Developer

- Graduated from Muscle Shoals High School, Muscle Shoals, Alabama.
- Pursuing a Computer Science degree from Auburn University.
- Desires to work in the field of computer science post graduation.

Evans Bowers - Developer

- Graduated from Boone High School in Orlando, FL
- Pursuing a Computer Science degree from Auburn University.
- Desires to work for an open-source based company post graduation.

Kaleb Sims - Developer

- Graduated from Wetumpka High School, Elmore, Alabama.
- Pursuing a Computer Engineering degree from Auburn University.
- Desires to work for a computer hardware company post graduation.

Project Overview

- Visualize air pollution’s severity and extent in China.
- Use data analysis and interactive maps and graphs.
- Highlight sources of pollution and impact on health.
- Show effectiveness of policies and regulations.
- Educate Chinese government and the public on urgency of action.

Air Pollution...

- Is contamination of air by chemical, physical, or biological agents
- Modifies the natural characteristic of the environment
- Negatively affects earth's climate and ecosystems globally

Air Pollution Gases

- The carbon oxides: carbon monoxide and carbon dioxide
- Nitrogen oxides and nitrogen dioxide
- Sulfur oxides and sulfur dioxide

What Causes Air Pollution?

- Energy use and production
- Transportation emissions
- Electricity and heating homes
- By products of manufacturing and burning coal
- Fumes from chemical production

Why is Air Pollution Bad?

Impact on Humans

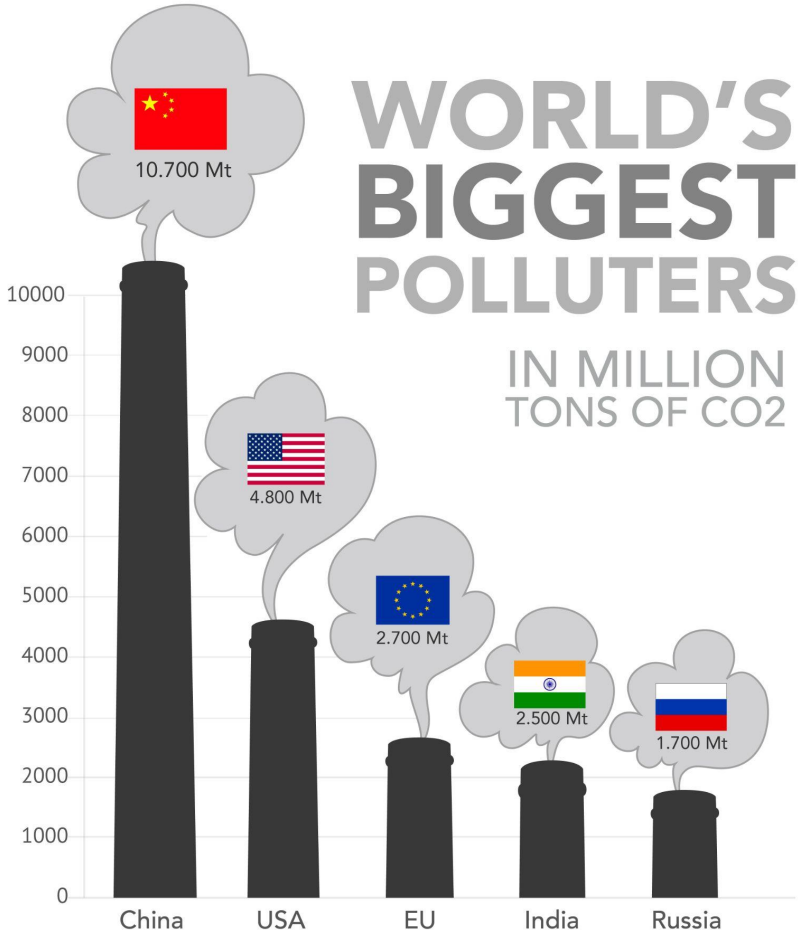
- When breathing in polluted air it enters lungs and bloodstream
- Shortness of breath, wheezing, coughing, chest pain, fatigue, sore throat
- Cardiovascular, heart disease, asthma, COPD

Why is Air Pollution Bad?

Impact on World

- Contaminates water and soil
- Kills plants and animals that rely on these resources
- Promotes global warming

China is one of the **BIGGEST** contributors to air pollution



China

- Located in East Asia spanning Kazakhstan to the Pacific Ocean.
- The most populous country with 1.4 billion people.
- The largest manufacturer and exporter in the world
- China has a major air pollution problem

China’s Air Pollution Problem

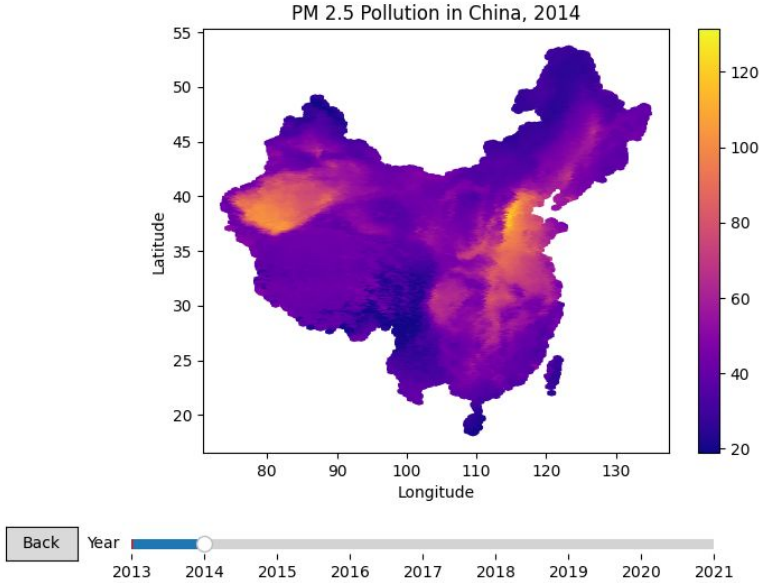
- Biggest contributors: burning coal in factories, power plants, and vehicles
- Citizens life expectancies are suffering from air pollution
- Pollution monitors readings reached 30-45 times recommended daily levels

China’s War Against Pollution

- Public concern rose in the late 1990’s
- Began the war in 2014
- National Air Quality Action Plan projected to improve quality by 2017
- Focus on combating the heaviest polluted areas first

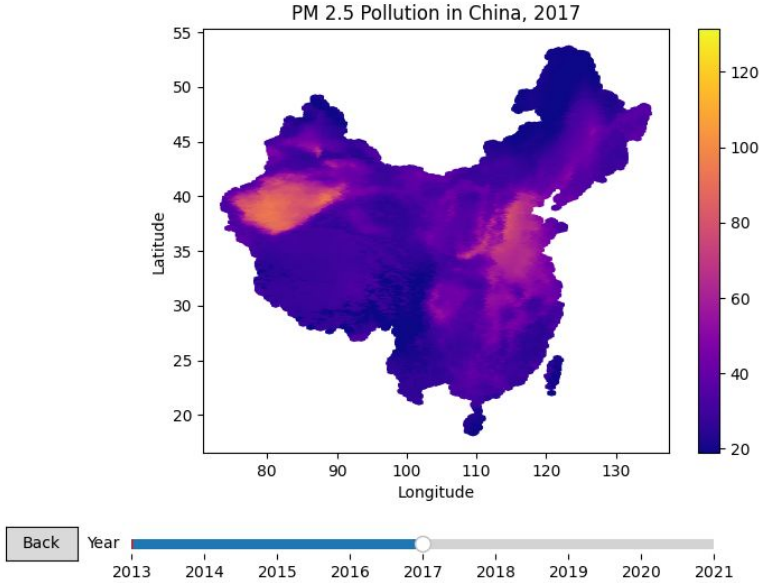
China’s Clean Air Act

- Cut down coal-power plants
- Increase renewable energy generation
- Reduce iron and steel making capacity in industry
- Declined by an average of 39.6 percent
- Has not been in top five since 2016



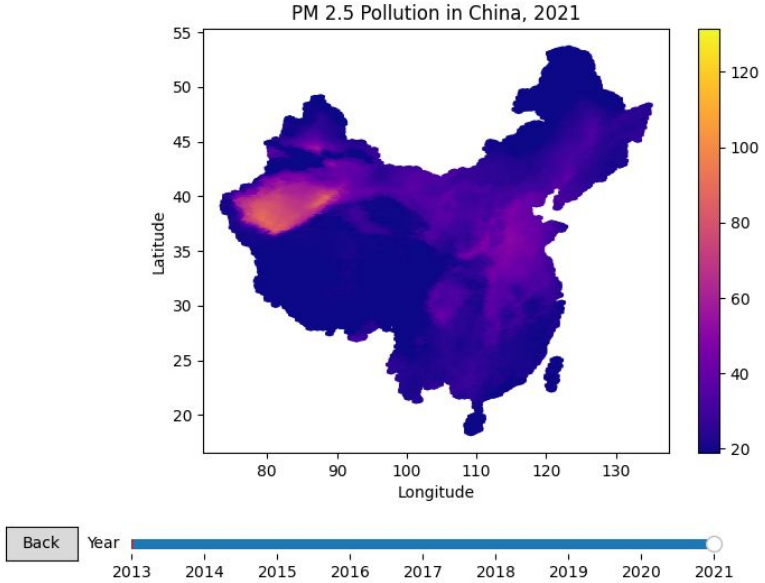
China’s Clean Air Act

- Cut down coal-power plants
- Increase renewable energy generation
- Reduce iron and steel making capacity in industry
- Declined by an average of 39.6 percent
- Has not been in top five since 2016



China’s Clean Air Act

- Cut down coal-power plants
- Increase renewable energy generation
- Reduce iron and steel making capacity in industry
- Declined by an average of 39.6 percent
- Has not been in top five since 2016



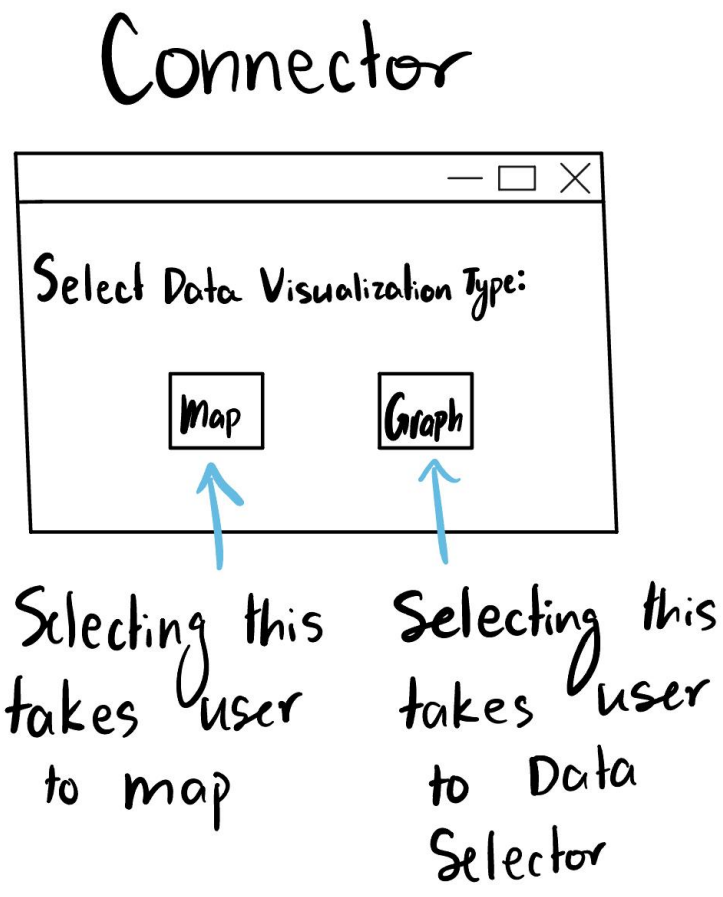
How it fits China’s CVDs

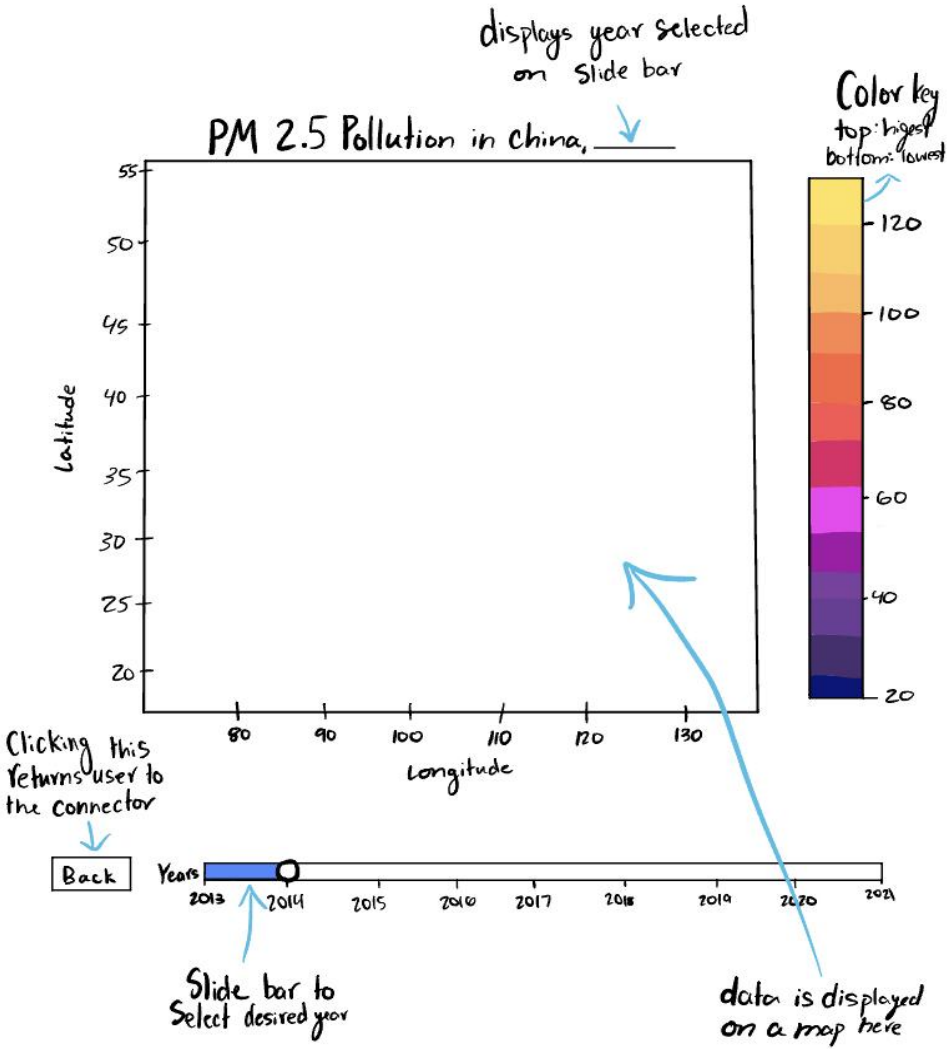
- Air pollution laws are long term plans
- The government enacts these laws fitting the high power distance

Goals of Project Code

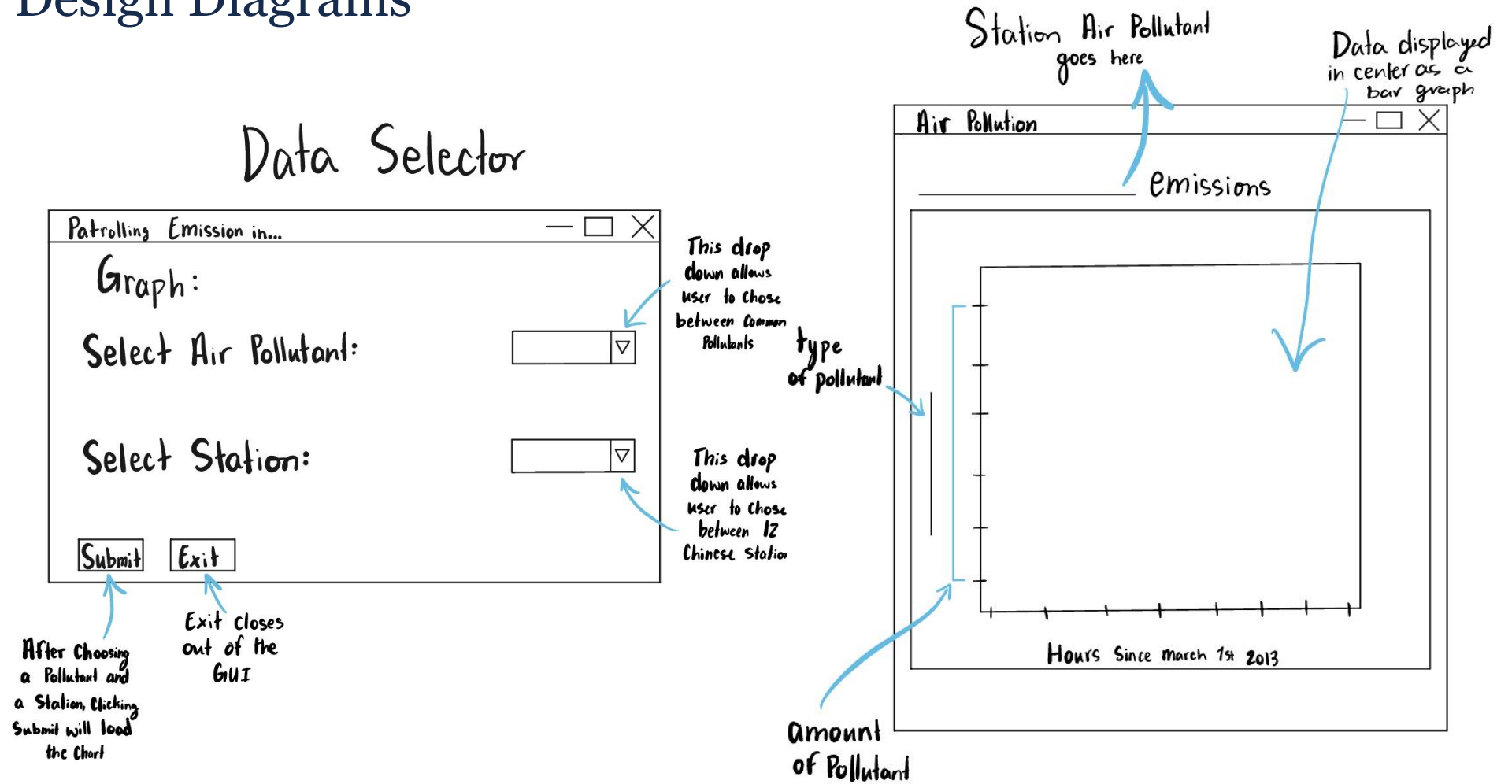
- Visualize air pollution severity and extent in China.
- Employ data analysis techniques.
- Develop interactive maps and graphs.

Design Diagrams





Design Diagrams



Demo

Progression of the graph

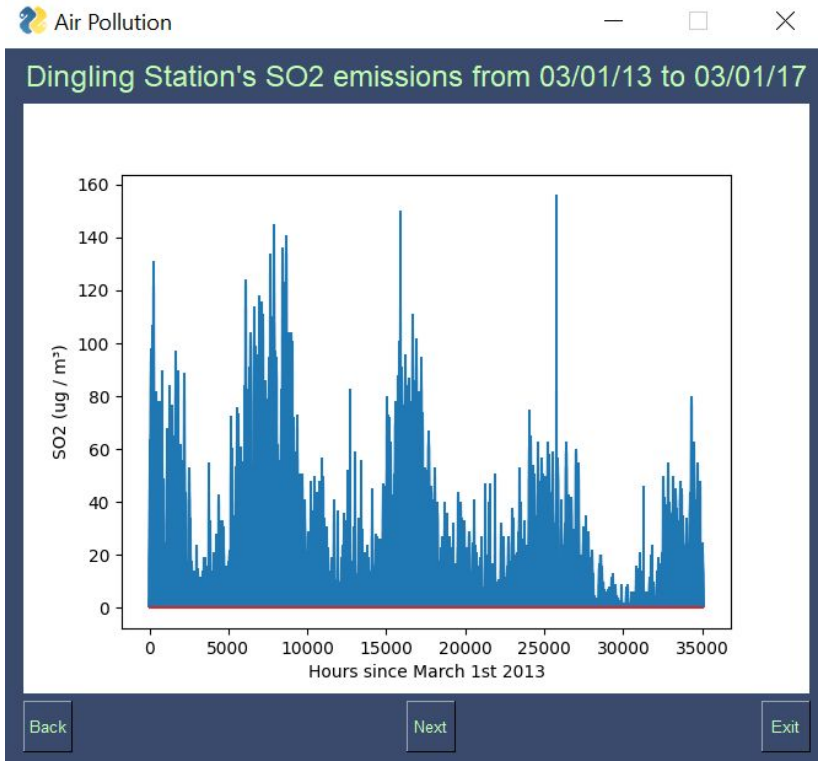
3/23 Formulated idea

3/28 Created original graph code

4/2 Optimized Code

4/6 Optimized Code

4/11 Added a back button



Progression of the interactive map

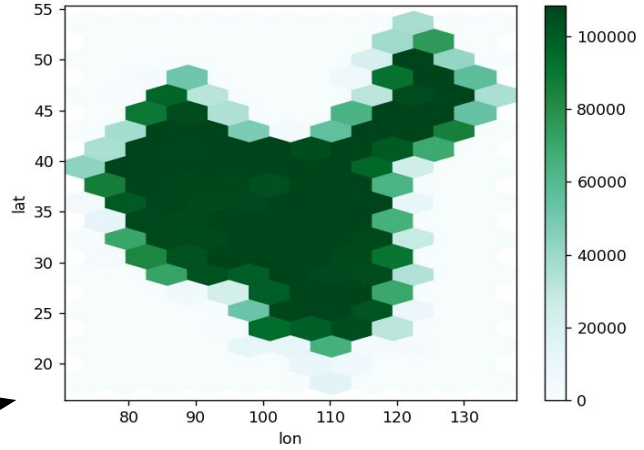
3/21 Formulated idea

3/30 Created blank graph plot

4/2 Added map w/ data plotted

4/6 Added functional sliding bar

4/13 Added a return button



Prototype from April 2

Progression of the interactive map

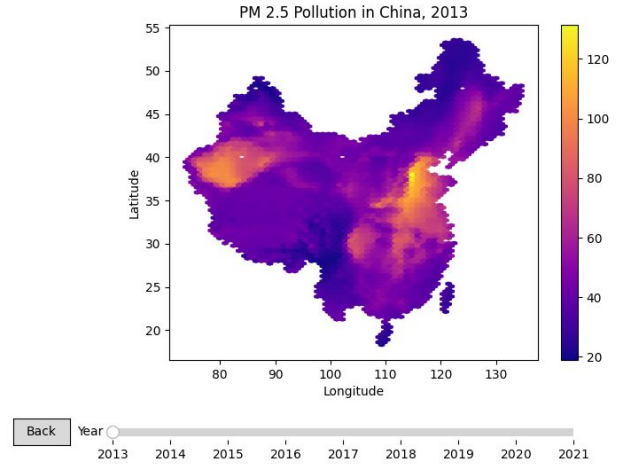
3/21 Formulated idea

3/30 Created blank graph plot

4/2 Added map w/ data plotted

4/6 Added functional sliding bar

4/13 Added a return button



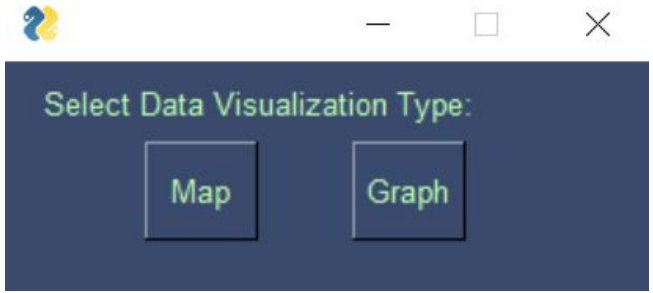
Current version as of April 13

Progression of the connector

4/4 Formulated idea

4/11 Created original connector code

4/11 Optimized Code



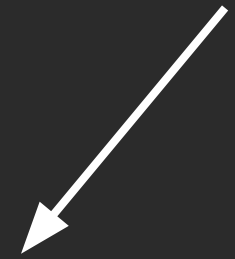
```python
import PySimpleGUI as sg
import matplotlib.pyplot as plt
import matplotlib
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
import pandas as pd
import os
from matplotlib.widgets import Slider, Button
import xarray as xr
```

```
def main():
    matplotlib.use('TkAgg')
    layout = [
        [sg.Text(f"Select Data Visualization Type:")],

        [sg.Button('Map', pad=((7, 10), 3), size=(5, 2)),
         sg.Button('Graph', pad=((7, 0), 3), size=(5, 2))]
    ]
    window = sg.Window('', layout, size=(300, 150))
    event, values = window.read()
    while event:
        if event == sg.WINDOW_CLOSE_ATTEMPTED_EVENT or event == 'Exit':
            exit()
        if event == 'Graph':
            window.close()
            del window
            os.system('maingraphs.py')
            break
        if event == 'Map':
            window.close()
            del window
            os.system('intmap.py')
            os.system('connector.py')
            break
```

Creates a GUI interface that accepts user input

```
def main():
    matplotlib.use('TkAgg')
    layout = [
        [sg.Text(f"Select Data Visualization Type:")],

        [sg.Button('Map', pad=((7, 10), 3), size=(5, 2)),
         sg.Button('Graph', pad=((7, 0), 3), size=(5, 2))]
    ]
    window = sg.Window('', layout, size=(300, 150))
    event, values = window.read()
    while event:
        if event == sg.WINDOW_CLOSE_ATTEMPTED_EVENT or event == 'Exit':
            exit()
        if event == 'Graph':
            window.close()
            del window
            os.system('maingraphs.py')
            break
        if event == 'Map':
            window.close()
            del window
            os.system('intmap.py')
            os.system('connector.py')
            break
```

Loop that assigns functions to the buttons



Andrew Goetz *

```
def create_map():
    datasets = {}
    for year in range(2013, 2022):
        filename = f"data/PM_2.5/CHAP_PM2.5_Y1K_{year}_V4.nc"
        df = xr.open_dataset(filename).to_dataframe().reset_index(drop=False).dropna()
        df = df.iloc[::500, :]
        datasets[year] = pd.DataFrame(df.values, columns=df.columns).dropna()

    fig, ax = plt.subplots(figsize=(8,6), num='PM 2.5 Pollution in China')
    init_yr = 2013
    df = datasets[init_yr]
    hb = df.plot.hexbin(x='lon', y='lat', ax=ax, C='PM2.5', gridsize=(60,60),
                        cmap='plasma', title=f'PM 2.5 Pollution in China, {init_yr}',
                        xlabel='Longitude', ylabel='Latitude')
    plt.subplots_adjust(left=0.25, bottom=0.25)
    yr_ax = fig.add_axes([0.17, 0.1, 0.65, 0.03])
    yr_slider = Slider(ax=yr_ax, label='Year', valmin=2013, valmax=2021,
                       valinit=2013, valstep=1)
    yr_slider.valtext.set_visible(False)
    yr_ax.add_artist(yr_ax.xaxis)
    yr_ticks = range(2013, 2022)
    yr_ax.set_xticks(yr_ticks)
    back_ax = fig.add_axes([0.03, 0.09, 0.08, 0.05])
    back = Button(back_ax, "Back")
```

Andrew Goetz *

```
def create_map():
    datasets = {}
    for year in range(2013, 2022):
        filename = f"data/PM_2.5/CHAP_PM2.5_Y1K_{year}_V4.nc"
        df = xr.open_dataset(filename).to_dataframe().reset_index(drop=False).dropna()
        df = df.iloc[::500, :]
        datasets[year] = pd.DataFrame(df.values, columns=df.columns).dropna()

    fig, ax = plt.subplots(figsize=(8,6), num='PM 2.5 Pollution in China')
    init_yr = 2013
    df = datasets[init_yr]
    hb = df.plot.hexbin(x='lon', y='lat', ax=ax, C='PM2.5', gridsize=(60,60),
                        cmap='plasma', title=f'PM 2.5 Pollution in China, {init_yr}',
                        xlabel='Longitude', ylabel='Latitude')
    plt.subplots_adjust(left=0.25, bottom=0.25)
    yr_ax = fig.add_axes([0.17, 0.1, 0.65, 0.03])
    yr_slider = Slider(ax=yr_ax, label='Year', valmin=2013, valmax=2021,
                       valinit=2013, valstep=1)
    yr_slider.valtext.set_visible(False)
    yr_ax.add_artist(yr_ax.xaxis)
    yr_ticks = range(2013, 2022)
    yr_ax.set_xticks(yr_ticks)
    back_ax = fig.add_axes([0.03, 0.09, 0.08, 0.05])
    back = Button(back_ax, "Back")
```

Loads datasets into pandas dataframe

Andrew Goetz *

```
def create_map():
    datasets = {}
    for year in range(2013, 2022):
        filename = f"data/PM_2.5/CHAP_PM2.5_Y1K_{year}_V4.nc"
        df = xr.open_dataset(filename).to_dataframe().reset_index(drop=False).dropna()
        df = df.iloc[::500, :]
        datasets[year] = pd.DataFrame(df.values, columns=df.columns).dropna()

    fig, ax = plt.subplots(figsize=(8,6), num='PM 2.5 Pollution in China')
    init_yr = 2013
    df = datasets[init_yr]
    hb = df.plot.hexbin(x='lon', y='lat', ax=ax, C='PM2.5', gridsize=(60,60),
                        cmap='plasma', title=f'PM 2.5 Pollution in China, {init_yr}',
                        xlabel='Longitude', ylabel='Latitude')
    plt.subplots_adjust(left=0.25, bottom=0.25)
    yr_ax = fig.add_axes([0.17, 0.1, 0.65, 0.03])
    yr_slider = Slider(ax=yr_ax, label='Year', valmin=2013, valmax=2021,
                       valinit=2013, valstep=1)
    yr_slider.valtext.set_visible(False)
    yr_ax.add_artist(yr_ax.xaxis)
    yr_ticks = range(2013, 2022)
    yr_ax.set_xticks(yr_ticks)
    back_ax = fig.add_axes([0.03, 0.09, 0.08, 0.05])
    back = Button(back_ax, "Back")
```

Creates hexbin plot from df

Andrew Goetz *

```
def create_map():
    datasets = {}
    for year in range(2013, 2022):
        filename = f"data/PM_2.5/CHAP_PM2.5_Y1K_{year}_V4.nc"
        df = xr.open_dataset(filename).to_dataframe().reset_index(drop=False).dropna()
        df = df.iloc[::500, :]
        datasets[year] = pd.DataFrame(df.values, columns=df.columns).dropna()

    fig, ax = plt.subplots(figsize=(8,6), num='PM 2.5 Pollution in China')
    init_yr = 2013
    df = datasets[init_yr]
    hb = df.plot.hexbin(x='lon', y='lat', ax=ax, C='PM2.5', gridsize=(60,60),
                        cmap='plasma', title=f'PM 2.5 Pollution in China, {init_yr}',
                        xlabel='Longitude', ylabel='Latitude')
    plt.subplots_adjust(left=0.25, bottom=0.25)
    yr_ax = fig.add_axes([0.17, 0.1, 0.65, 0.03])
    yr_slider = Slider(ax=yr_ax, label='Year', valmin=2013, valmax=2021,
                       valinit=2013, valstep=1)
    yr_slider.valtext.set_visible(False)
    yr_ax.add_artist(yr_ax.xaxis)
    yr_ticks = range(2013, 2022)
    yr_ax.set_xticks(yr_ticks)
    back_ax = fig.add_axes([0.03, 0.09, 0.08, 0.05])
    back = Button(back_ax, "Back")
```

Adds slider and back button

```python
def main_graph_gui():

    width = max(map(len, selection)) + 1
    matplotlib.use('TkAgg')

    # define GUI elements and window
    layout = [
        [sg.Text('Graph:')],

        [sg.Text('Select Air Pollutant:', size=(17, 1)),
         sg.Combo(types, size=(width, 5), enable_events=False, key='-type-')],

        [sg.Text('Select Station:', size=(17, 1)),
         sg.Combo(selection, size=(width, 5), enable_events=False, key='-city-')],

        [sg.Button('Back', size=(4, 2)), sg.Submit(), sg.Button('Exit')]
    ]
    window = sg.Window('Patrolling Emission in China', layout, size=(330, 120))
    event, values = window.read()

    if event == sg.WINDOW_CLOSE_ATTEMPTED_EVENT or event == 'Exit':
        exit()
    if event == 'Back':
        window.close()
        os.system('python connector.py')
        exit()

    airp = values['-type-']
    city = values['-city-']

    # close selection window and draw graph
    window.close()
    create_graph(city, airp)
```

```
def main_graph_gui():

    width = max(map(len, selection)) + 1
    matplotlib.use('TkAgg')

    # define GUI elements and window
    layout = [
        [sg.Text('Graph:')],

        [sg.Text('Select Air Pollutant:', size=(17, 1)),
         sg.Combo(types, size=(width, 5), enable_events=False, key='-type-')],

        [sg.Text('Select Station:', size=(17, 1)),
         sg.Combo(selection, size=(width, 5), enable_events=False, key='-city-')],

        [sg.Button('Back', size=(4, 2)), sg.Submit(), sg.Button('Exit')]
    ]
    window = sg.Window('Patrolling Emission in China', layout, size=(330, 120))
    event, values = window.read()

    if event == sg.WINDOW_CLOSE_ATTEMPTED_EVENT or event == 'Exit':
        exit()
    if event == 'Back':
        window.close()
```

Creates a GUI interface that accepts user input

```
     sg.Combo(types, size=(width, 5), enable_events=False, key='-type-')],

    [sg.Text('Select Station:', size=(17, 1)),
     sg.Combo(selection, size=(width, 5), enable_events=False, key='-city-')],

    [sg.Button('Back', size=(4, 2)), sg.Submit(), sg.Button('Exit')]
]
window = sg.Window('Patrolling Emission in China', layout, size=(330, 120))
event, values = window.read()

if event == sg.WINDOW_CLOSE_ATTEMPTED_EVENT or event == 'Exit':
    exit()
if event == 'Back':
    window.close()
    os.system('python connector.py')
    exit()

airp = values['-type-']
city = values['-city-']

# close selection window and draw graph
window.close()
create_graph(city, airp)
```

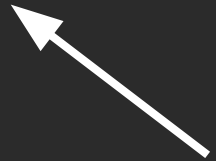
Assigns functions to the buttons

```
     sg.Combo(types, size=(width, 5), enable_events=False, key='-type-')],

    [sg.Text('Select Station:', size=(17, 1)),
     sg.Combo(selection, size=(width, 5), enable_events=False, key='-city-')],

    [sg.Button('Back', size=(4, 2)), sg.Submit(), sg.Button('Exit')]
]
window = sg.Window('Patrolling Emission in China', layout, size=(330, 120))
event, values = window.read()

if event == sg.WINDOW_CLOSE_ATTEMPTED_EVENT or event == 'Exit':
    exit()
if event == 'Back':
    window.close()
    os.system('python connector.py')
    exit()
```

Assigns values from user input

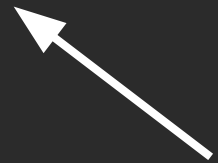
```
airp = values['-type-']
city = values['-city-']

# close selection window and draw graph
window.close()
create_graph(city, airp)
```

```
def make_graph(city, airp):

    file = pd.read_csv(f'Graphdata/PRSA_Data_{city}_20130301-20170228.csv')

    x_axis = file['No']
    y_axis = file[airp]

    plt.stem(x_axis, y_axis, markerfmt=" ")
    plt.xlabel("Hours since March 1st 2013")
    plt.ylabel(f'{airp} (ug / m\u00b3)')

    return plt.gcf()
```



Reads data file

```
def make_graph(city, airp):

    file = pd.read_csv(f'Graphdata/PRSA_Data_{city}_20130301-20170228.csv')

    x_axis = file['No']
    y_axis = file[airp]

    plt.stem(x_axis, y_axis, markerfmt=" ")
    plt.xlabel("Hours since March 1st 2013")
    plt.ylabel(f'{airp} (ug / m\u00b3)')

    return plt.gcf()
```



Creates graphs using user input

```
def create_graph(city, airp):
    if city == "All":

        for i in range(1, 13):

            cityname = selection[i]

            plot = make_graph(cityname, airp)

            create_all_graphs_gui(plot, cityname, airp)
            plt.clf()
    else:

        plot = make_graph(city, airp)

        create_graph_gui(plot, city, airp)
        plt.clf()
```

Runs other functions

```
def create_graph(city, airp):
    if city == "All":

        for i in range(1, 13):

            cityname = selection[i]

            plot = make_graph(cityname, airp)

            create_all_graphs_gui(plot, cityname, airp)
            plt.clf()
    else:

        plot = make_graph(city, airp)

        create_graph_gui(plot, city, airp)
        plt.clf()
```

Runs other functions

```
def create_all_graphs_gui(fig, cityname, airp):

    figure_x, figure_y, figure_w, figure_h = fig.bbox.bounds
    layout = [
                [sg.Text(f"{cityname} Station's {airp} emissions from 03/01/13 to 03/01/17", font='Any 18')],

                [sg.Canvas(size=(figure_w, figure_h), key='canvas')],

                [sg.Button('Back', size=(4, 2)),
                 sg.Button('Next', pad=((figure_w / 2.4, 0), 2), size=(4, 2)),
                 sg.Button('Exit', pad=((figure_w / 2.57, 0), 2), size=(4, 2))]
             ]

    window = sg.Window('Air Pollution', layout, force_toplevel=True,
                       finalize=True, enable_close_attempted_event=True)
    draw_figure(window['canvas'].TKCanvas, fig)
    event, values = window.read()

    while event:

        if event == 'Back':
            window.close()
            plt.clf()
            del window
            main_graph_gui()
            break

        if event == sg.WINDOW_CLOSE_ATTEMPTED_EVENT or event == 'Exit':
            exit()

        if event == 'Next':
            plt.clf()
            window.close()
            break
```

```
def create_all_graphs_gui(fig, cityname, airp):

    figure_x, figure_y, figure_w, figure_h = fig.bbox.bounds
    layout = [
                [sg.Text(f"{cityname} Station's {airp} emissions from 03/01/13 to 03/01/17", font='Any 18')],

                [sg.Canvas(size=(figure_w, figure_h), key='canvas')],

                [sg.Button('Back', size=(4, 2)),
                 sg.Button('Next', pad=((figure_w / 2.4, 0), 2), size=(4, 2)),
                 sg.Button('Exit', pad=((figure_w / 2.57, 0), 2), size=(4, 2))]
             ]

    window = sg.Window('Air Pollution', layout, force_toplevel=True,
                       finalize=True, enable_close_attempted_event=True)
    draw_figure(window['canvas'].TKCanvas, fig)
    event, values = window.read()

    while event:

        if event == 'Back':
            window.close()
            plt.clf()
            del window
            main_graph_gui()
            break

        if event == sg.WINDOW_CLOSE_ATTEMPTED_EVENT or event == 'Exit':
```

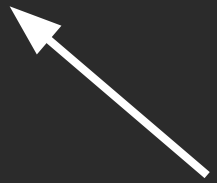
Creates the GUI for all of the stations

```
            [sg.Button('Back', size=(4, 2)),
             sg.Button('Next', pad=((figure_w / 2.4, 0), 2), size=(4, 2)),
             sg.Button('Exit', pad=((figure_w / 2.57, 0), 2), size=(4, 2))]
        ]

window = sg.Window('Air Pollution', layout, force_toplevel=True,
                   finalize=True, enable_close_attempted_event=True)
draw_figure(window['canvas'].TKCanvas, fig)
event, values = window.read()
```

Loop that assigns functions to the buttons

```
while event:

    if event == 'Back':
        window.close()
        plt.clf()
        del window
        main_graph_gui()
        break

    if event == sg.WINDOW_CLOSE_ATTEMPTED_EVENT or event == 'Exit':
        exit()

    if event == 'Next':
        plt.clf()
        window.close()
        break
```

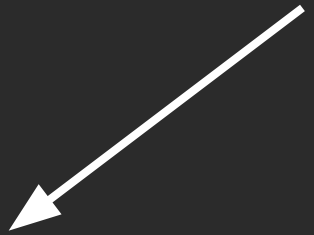
```
def create_graph_gui(fig, city, airp):
    figure_x, figure_y, figure_w, figure_h = fig.bbox.bounds

    layout = [
                [sg.Text(f"{city} Station's {airp} emissions from 03/01/13 to 03/01/17", font='Any 18')],

                [sg.Canvas(size=(figure_w, figure_h), key='canvas')],

                [sg.Button('Back', size=(4, 2)),
                 sg.Button('Exit', pad=(((figure_w / 1.152), 0), 3), size=(4, 2))]
            ]

    window = sg.Window('Air Pollution', layout, force_toplevel=True,
                       finalize=True, enable_close_attempted_event=True)
    draw_figure(window['canvas'].TKCanvas, fig)
    event, values = window.read()

    while event:

        if event == 'Back':
            window.close()
            plt.clf()
            del window
            main_graph_gui()
            break

        elif event == sg.WINDOW_CLOSE_ATTEMPTED_EVENT or event == 'Exit':
            exit()
```



Creates the GUI for one of the stations

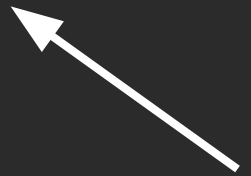
```
def create_graph_gui(fig, city, airp):

    figure_x, figure_y, figure_w, figure_h = fig.bbox.bounds

    layout = [
                [sg.Text(f"{city} Station's {airp} emissions from 03/01/13 to 03/01/17", font='Any 18')],

                [sg.Canvas(size=(figure_w, figure_h), key='canvas')],

                [sg.Button('Back', size=(4, 2)),
                 sg.Button('Exit', pad=(((figure_w / 1.152), 0), 3), size=(4, 2))]
            ]

    window = sg.Window('Air Pollution', layout, force_toplevel=True,
                       finalize=True, enable_close_attempted_event=True)
    draw_figure(window['canvas'].TKCanvas, fig)
    event, values = window.read()
```

Loop that assigns functions to the buttons



```
    while event:

        if event == 'Back':
            window.close()
            plt.clf()
            del window
            main_graph_gui()
            break

        elif event == sg.WINDOW_CLOSE_ATTEMPTED_EVENT or event == 'Exit':
            exit()
```

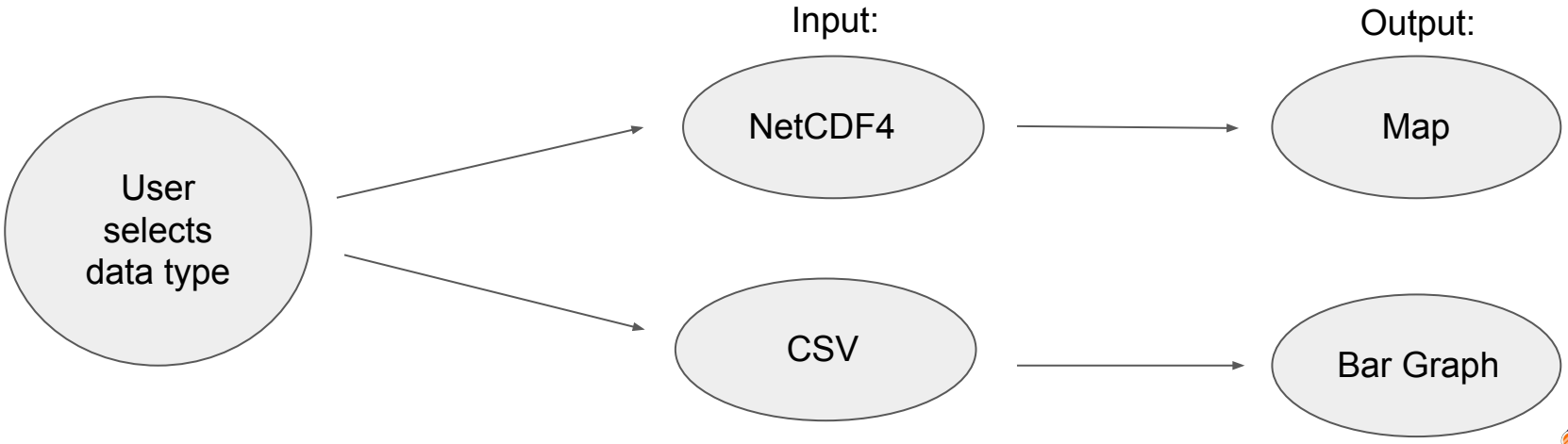
```
def draw_figure(canvas, figure):

    figure_canvas_agg = FigureCanvasTkAgg(figure, canvas)
    figure_canvas_agg.draw()
    figure_canvas_agg.get_tk_widget().pack(side='top', fill='both', expand=1)
```



Converts matplotlib graph into a GUI figure

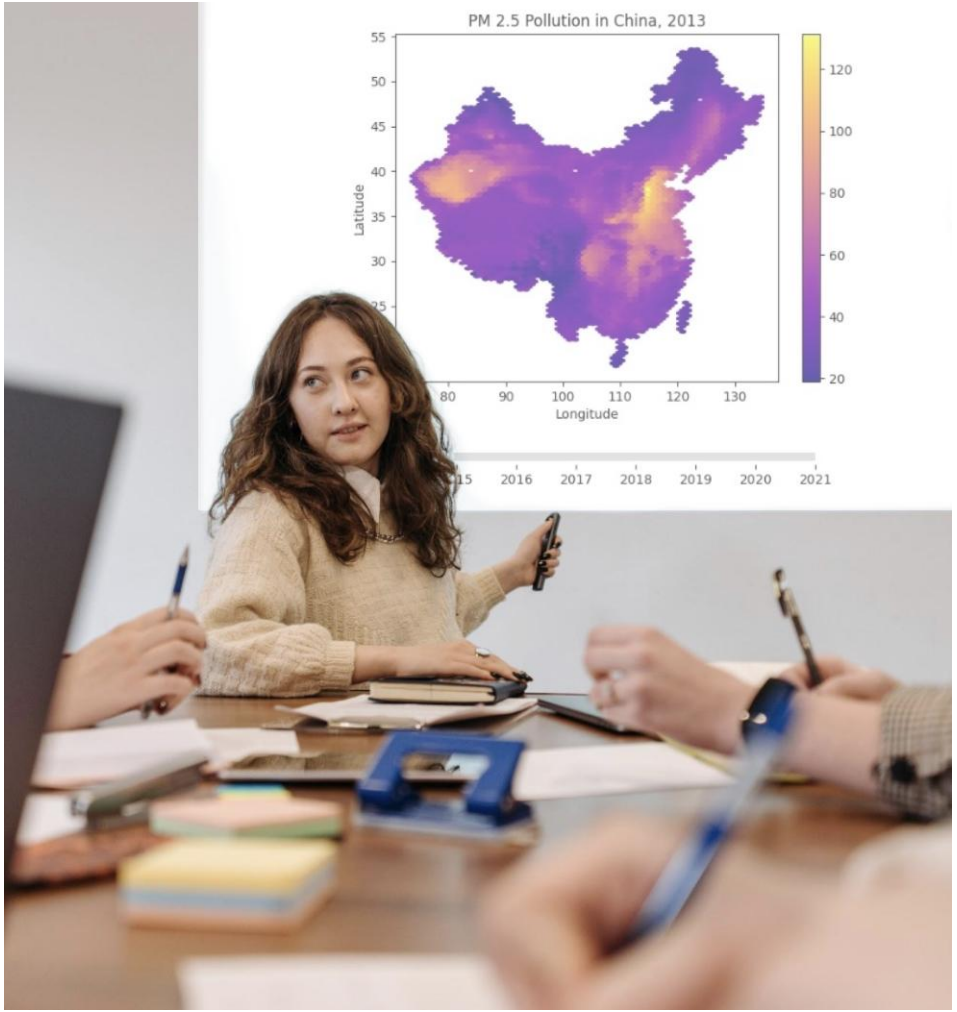
Data flow/inputs

- Inputs are from the csv and NetCDF4 files.
- NetCDF4 gives geographical data for the map
- csv files give data for the bar graphs
- Files are opened after the data is selected by the user



Our product's usage

1) Government
   - highlight key areas of focus
   - decide where to allocate resources
2) Landowners
   - estimating property value
3) Home buyers
   - deciding where to live



What did we accomplish?

- Created products to provide valuable insights
- Implemented a variety of tools in Python
- Gained a better understanding of Chinese culture
- Worked with effective team communication

What would we add/improve with more time?

- Insights on different kinds of pollution (CO, NO2)
- More details on the map (specific cities or factories)
- Ability to select different pollutants on the map
- The speed and quality of the product

Sources

Data:

- Wei, Jing. “ChinaHighAirPollutants (CHAP).” *Github*, https://weijing-rs.github.io/product.html.
- “China Air Pollution.” Kaggle, https://www.kaggle.com/competitions/chinaairpollution/data.

Other resources:

- “Aqli Policy Impacts-China: National Air Quality Action Plan (2014).” AQLI, 15 June 2022, https://aqli.epic.uchicago.edu/policy-impacts/china-national-air-quality-action-plan-2014/.
- Upton-McLaughlin, Sean. “Proper Character and Behavior.” The China Culture Corner, 20 Feb. 2020, https://chinaculturecorner.com/2013/06/05/proper-character-in-china/.
- “Aqli Policy Impacts-China: National Air Quality Action Plan (2014).” *AQLI*, 15 June 2022, https://aqli.epic.uchicago.edu/policy-impacts/china-national-air-quality-action-plan-2014/#:~:text=Yet%2C%20it%20hasn't%20been,taken%20after%20intense%20public%20scrutiny.


THANK YOU!
谢谢