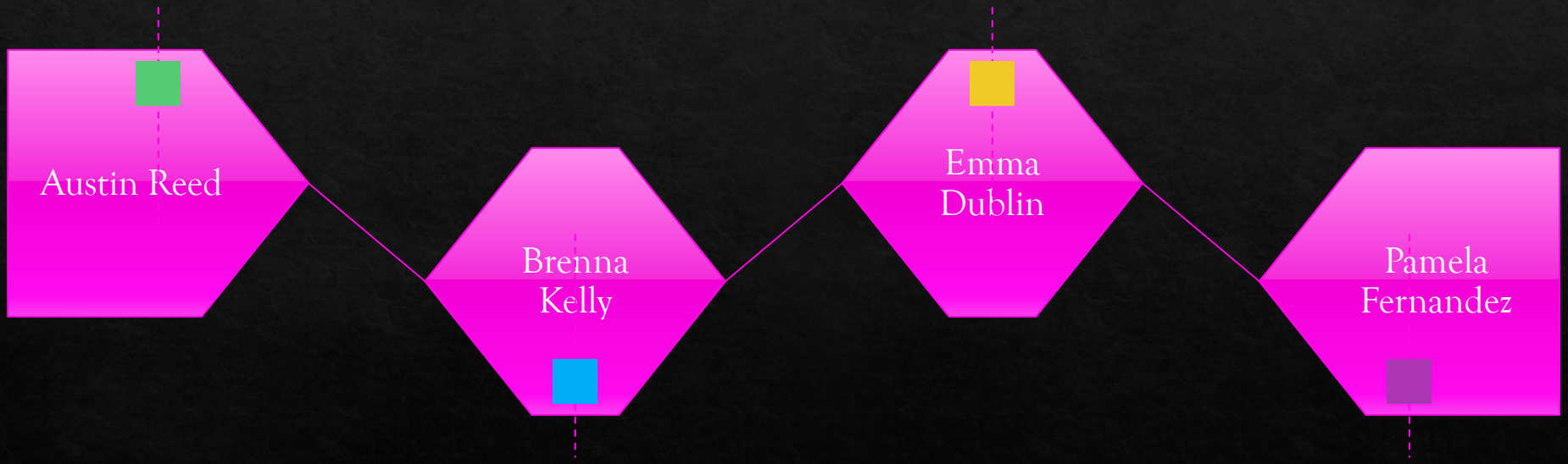

Pixel Wars-
r/place™ Canvas
Collaboration
A reddit April Fool's Data
Visualization Analysis

Group Three Contributors



Objectives

Inspiration

Data

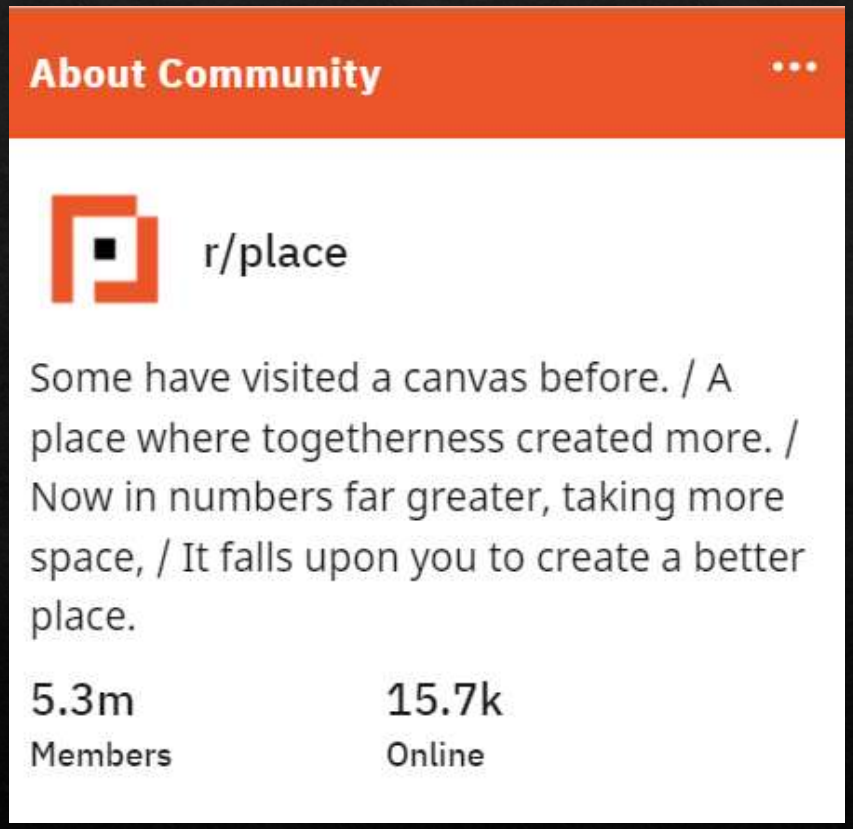
Tools

Process

Conclusion

Resources

Inspiration



https://styles.redditmedia.com/t5_2sxhs/styles/image_widget_niyjd7soesq81.png

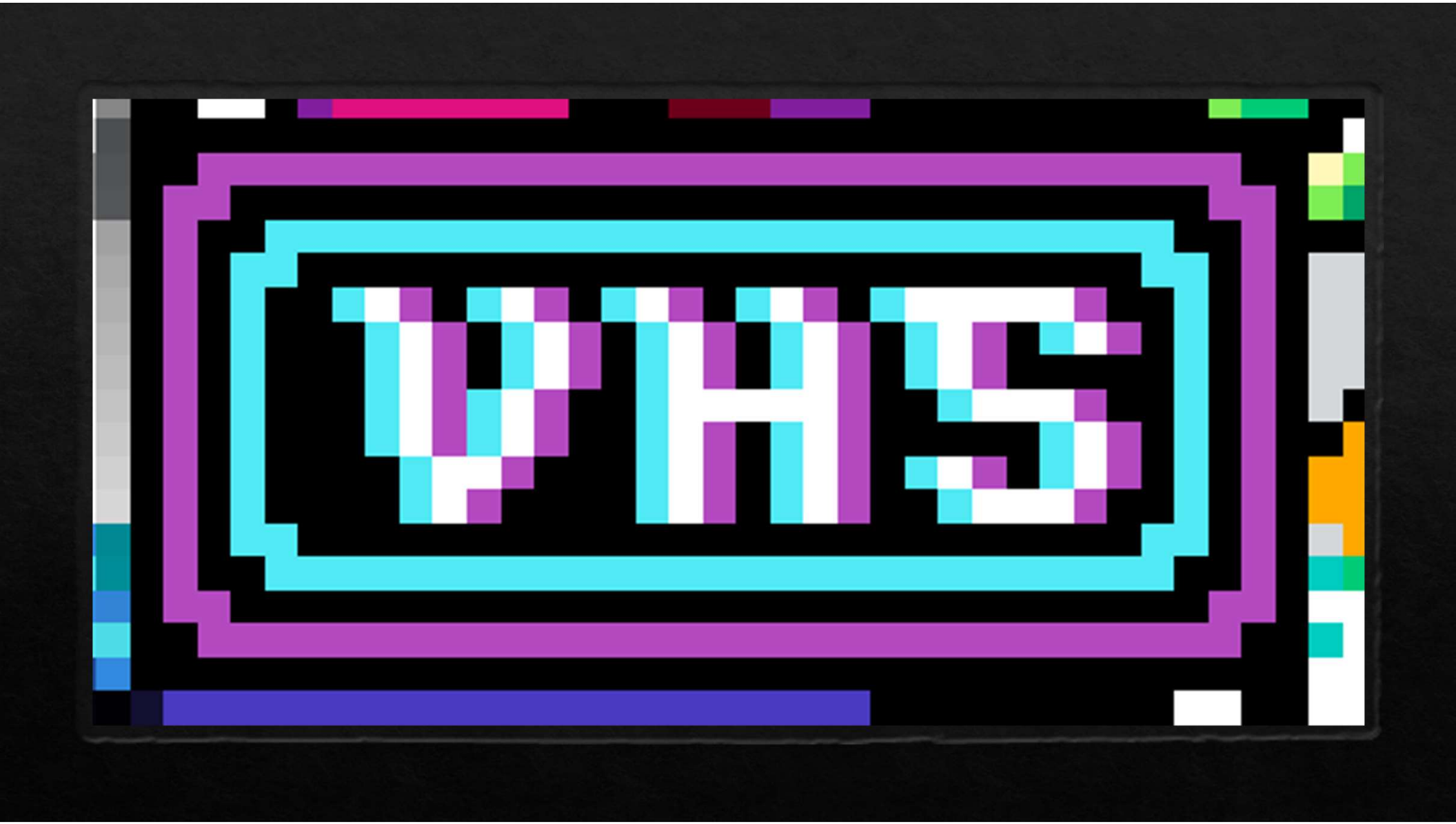
The Canvas

Source: https://www.youtube.com/watch?v=-JUZtbhs5DE

Color Palette

https://www.reddit.com/r/place/comments/txowoa/a_rplace_2022_final_palette_high_resolution_hex/

Canada's Evolution on /r/place


VHS

Data Manipulation

- Obtained data
- Transformed data
- Wrote a program to import data to SQL
- Created queries in the database to select data we wanted to visualize
- Executed the queries with SQLAlchemy
- Convert the data into json arrays for D3 and plotly
- Defined the functions needed to create the plots
- Created a flask app to send the code to our webpage
- Create the html code and formatting - include bootstrap and a css file

Data Overview

~80 files with approximately 2 million rows of data each.

/reddit and Kaggle datasets

Google API – JSON
Google Restaurant Reviews

Data Output | Explain | Messages | Notifications

	id [PK] integer	timestamp timestamp without time zone	user_id character varying	pixel_color character varying	x integer	y integer
1	1	2022-04-04 00:53:51.577	ovTZk4GyTS1mDQnTbV+vDOCu1f+u6w+CkIZ6445vD4XN8aIFy/6GtNkYp5MSic6Tjo/fBCCGe6oZKMAN3rEZHw==	#00CCC0	826	1048
2	2	2022-04-04 00:53:53.758	6NSgFa1CvIPly1VniNhlbrmoN3vgDFbMSKqh+c4TTfrr3dMib91oUWONX96g5PPciolxedF24ldNOu/g5yqDrg==	#94B3FF	583	1031
3	3	2022-04-04 00:53:54.685	O5Otyp3Z3owzTuwM9XnMggpLcqKEumsOMKGhRiDTTlmWbNLhLKmLl4gn1QPbaABqZqmFC/OmE/0732n39dGIQ==	#6A5CFF	1873	558
4	4	2022-04-04 00:54:57.541	tc273UiqS0wKa6VwiOs/iz/t4LyPYrhL2Q347awn11IQQELrEzZBCmGe28NWM+O1IdfH4CieCpEnE5sHecW90w==	#009EAA	1627	255
5	5	2022-04-04 00:55:16.307	OOWsU/HLb4UUkQwclDeXFtsJTOXMlAdNHiRpFA1Qk+SxUrJE7lpGFevfV9w+zlmFimmNANlDdfN3kluz69M9MQ==	#94B3FF	49	1478
6	6	2022-04-04 00:55:20.64	A0HdtcPvI7ipKivvXNVZDa3gkcjGXFjNxF5tca5QXazENCGVR8d3j65QVgnaVlgvGbdtiGuSvRs1rAj2f1oMAQ==	#E4ABFF	408	1863
7	7	2022-04-04 00:55:34.898	1U4LPuB22P6Yf7eRhKz6zU1dFMK5wXlzsNVPUNhP7eHlwGuMfnDz/hDh/8QhDug6qqsKXHYvgH9L5FnlWNMGm3A==	#94B3FF	111	1582
8	8	2022-04-04 00:55:40.375	0AGoMGF50j0DJDc+704SwMylU90YDILDlgo8WOetgpiEWGEhMB3Eh8Q8r+Wa9XlJhQZQcOquqpRgZ5REB4jrOA==	#6A5CFF	1334	1840
9	9	2022-04-04 00:55:57.168	tPcrtm7OtEmSThdRSWmB7jmTF9lUVZ1pltNv1oKqPY9bom/EGlO3/b5kjRenbD3vMF48psnR9MnhlrTT1bpC9A==	#6A5CFF	1908	1854
10	10	2022-04-04 00:56:07.43	7wfv1Tew3St1xYXQrTy7EF4LYwuuzxb0TXOxzxGBXriim81odvA5tXi5MQspC3K2u4rO5Ns6dUWJ/Lhheleb8A==	#009EAA	1504	1722
11	11	2022-04-04 00:56:32.636	ib4u+ORY5n4gMEfn9hXET9u5Smqf35XpDsT0LSw8uczU4gX5ObeKxW2nsnbcYC2Pfcw+26uyUjaMrytzkIT6hg==	#94B3FF	1850	1809

Database Generator

```python
import pandas as pd
from sqlalchemy import create_engine, Column, Integer, String, ForeignKey, DateTime
from sqlalchemy.orm import declarative_base
import psycopg2

table_name = 'pixel_placement'

Base = declarative_base()

class Pixel_placement(Base):
    __tablename__ = table_name

    id = Column(Integer, primary_key=True)
    timestamp = Column(DateTime)
    user_id = Column(String)
    pixel_color = Column(String)
    x = Column(Integer)
    y = Column(Integer)

conn_string = 'postgresql+psycopg2://postgres:pass@127.0.0.1/reddit'
db = create_engine(conn_string)

conn = psycopg2.connect("dbname=reddit user=postgres password=admin")
conn.autocommit = True
cursor = conn.cursor()

Base.metadata.create_all(db)

prefix = 'https://placedata.reddit.com/data/canvas-history/2022_place_canvas_history-0000000000'
extension = '.csv.gzip'
for i in range(0, 79):
    url = f'{prefix}{str(i).zfill(2)}{extension}'
    print(f'Downloading file {url}...      ', end='', flush=True)
    df = pd.read_csv(url, compression='gzip')

    print(f'Processing...      ', end='', flush=True)
    df['x'] = df['coordinate'].str.split(',').str[0]
    df['y'] = df['coordinate'].str.split(',').str[1]
    df = df.drop('coordinate', axis=1)

    print(f'Loading into database...      ', end='', flush=True)
    df.to_sql(table_name, db, if_exists='append', index=False)
    conn.commit()
    print(f'Done')
```

```
❯ python generate_database.py
Downloading file https://placedata.reddit.com/data/canvas-history/2022_place_canvas_history-000000000000.csv.gzip...    Processing...    Loading into database...    Done
Downloading file https://placedata.reddit.com/data/canvas-history/2022_place_canvas_history-000000000001.csv.gzip...    Processing...    Loading into database...    Done
Downloading file https://placedata.reddit.com/data/canvas-history/2022_place_canvas_history-000000000002.csv.gzip...    Processing...    Loading into database...    Done
Downloading file https://placedata.reddit.com/data/canvas-history/2022_place_canvas_history-000000000003.csv.gzip...    Processing...    Loading into database...    Done
Downloading file https://placedata.reddit.com/data/canvas-history/2022_place_canvas_history-000000000004.csv.gzip...    Processing...    Loading into database...    Done
Downloading file https://placedata.reddit.com/data/canvas-history/2022_place_canvas_history-000000000005.csv.gzip...    Processing...    Loading into database...    Done
Downloading file https://placedata.reddit.com/data/canvas-history/2022_place_canvas_history-000000000006.csv.gzip...    Processing...    Loading into database...    Done
Downloading file https://placedata.reddit.com/data/canvas-history/2022_place_canvas_history-000000000007.csv.gzip...    Processing...    Loading into database...    Done
```

Data Analysis/Renderings

Our HTML Page

Pixel Wars

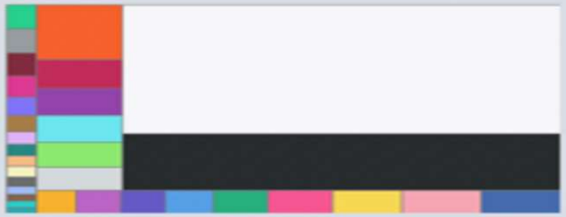
visualizations from the r/place 2022 canvas

Location To Analyze

Dataset1 Name Here

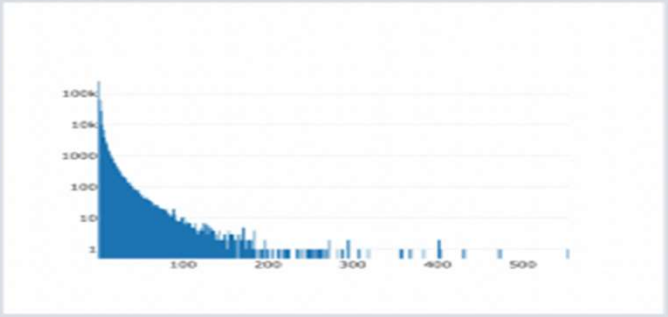
Colors Used

a proportional representation of the colors used



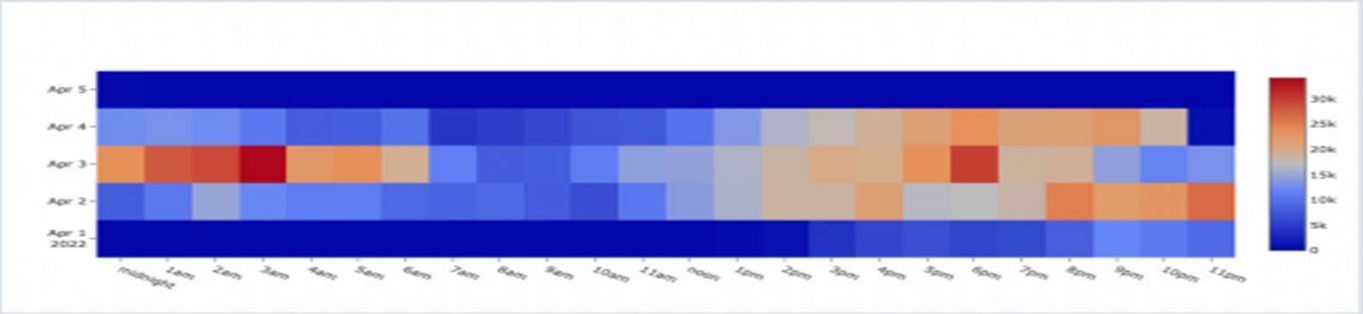
Pixels by User

a logarithmic representation of the number of people who placed the given number of pixels

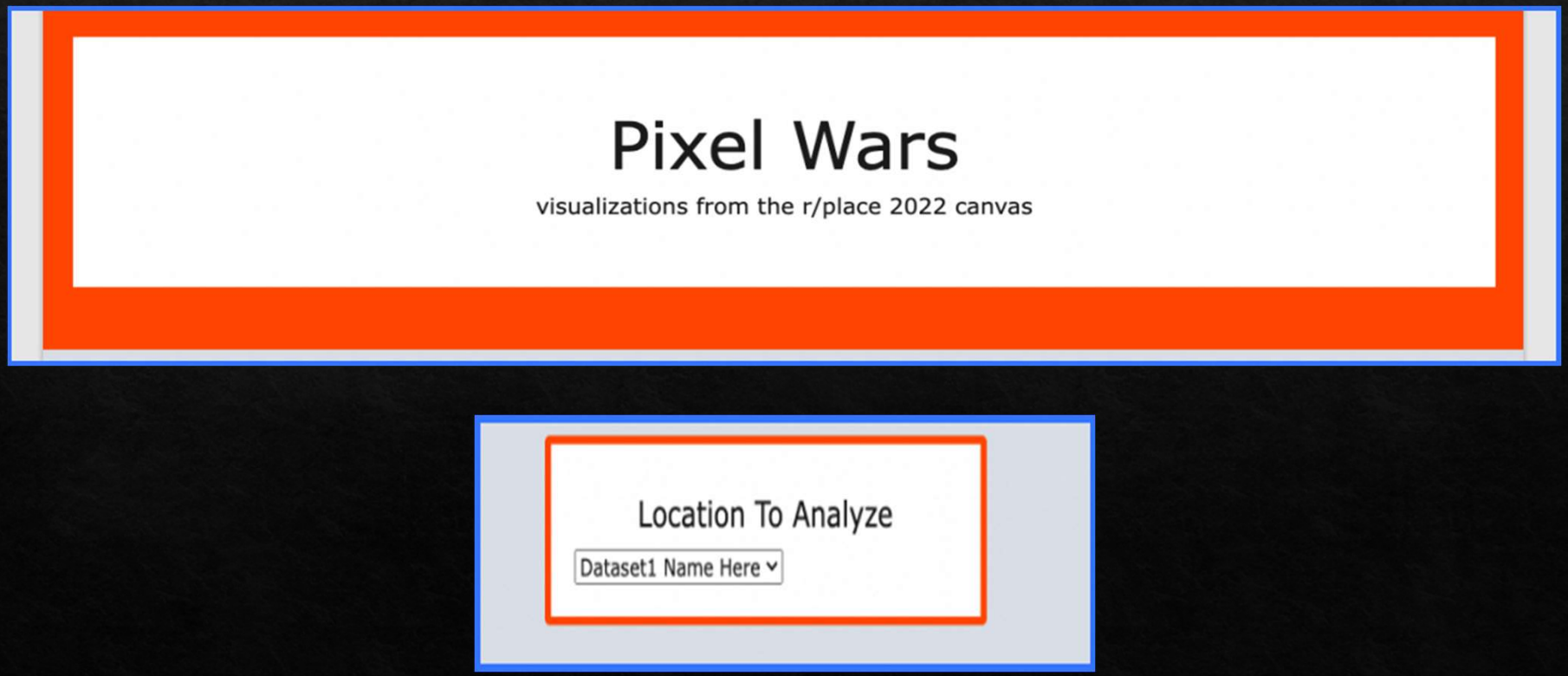


Heatmap of Pixels Placed by Date and Time

a graphical representation of data the most active times users were placing pixels



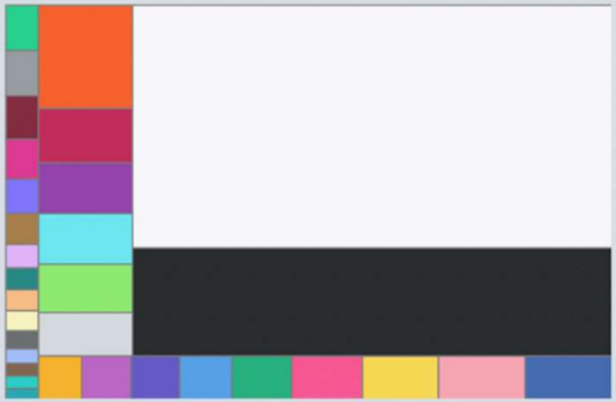
Page Elements



Page Elements

Colors Used

a proportional representation of the colors used

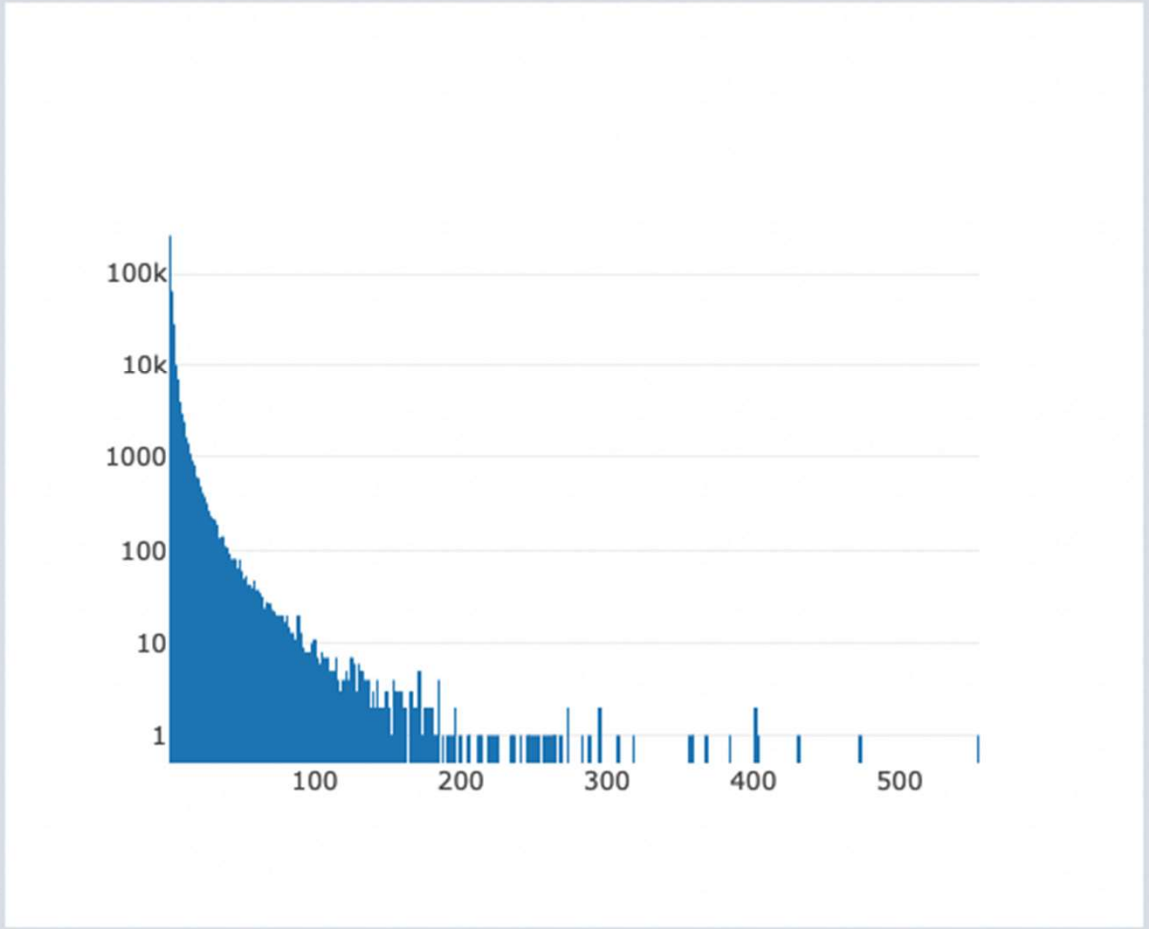


```
import squarify

# squareify plot showing the number of times each color was used - wedge color is the color used


sizes = colors_result[1]

colors= colors_result[0]


plot = squarify.plot(sizes, color=colors, alpha=.8, edgecolor='gray')

plt.axis('off')
```

Page Elements

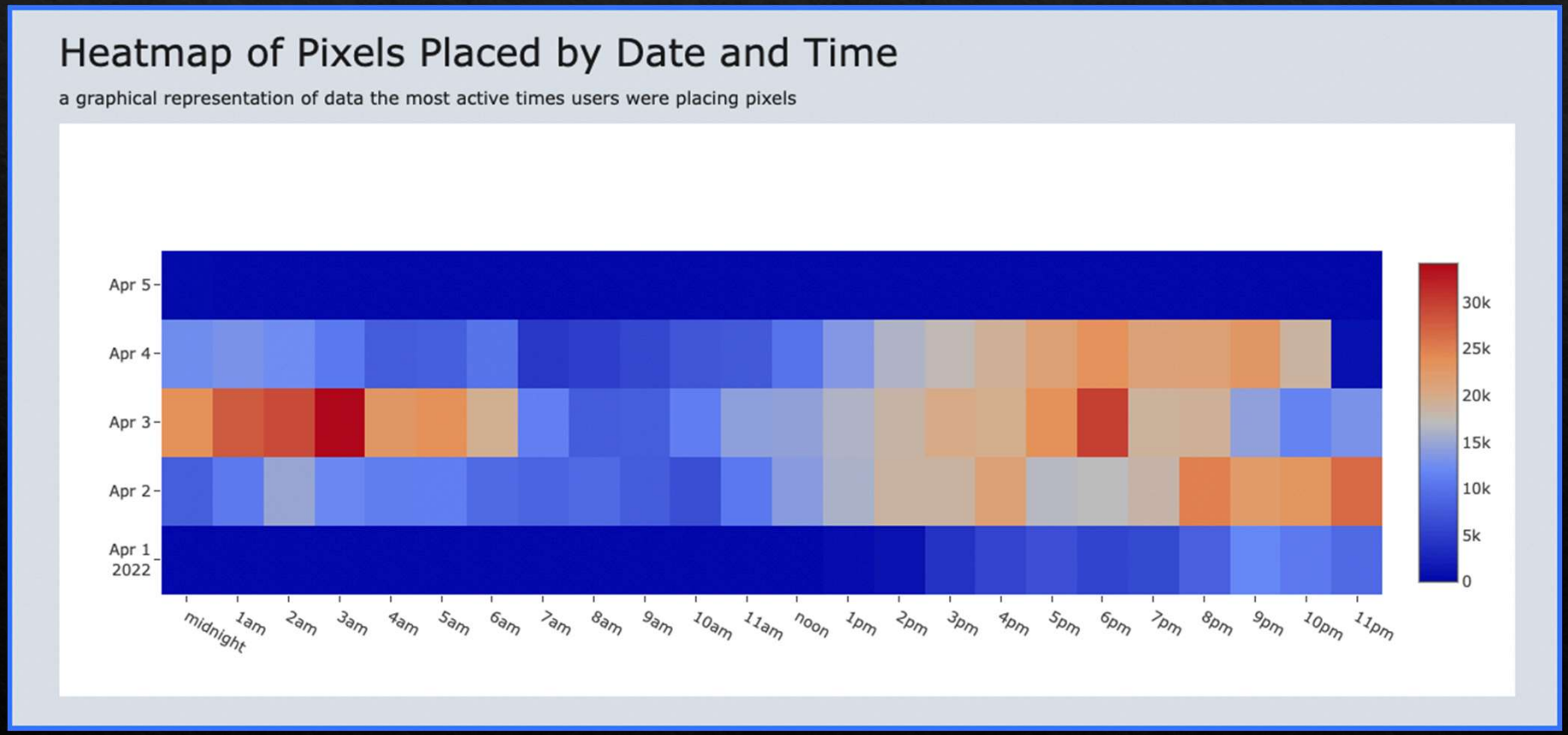
Pixels by User

a logarithmic representation of the number of people who placed the given number of pixels



```python
import plotly
import plotly.graph_objs as go
import plotly.express as px


import pandas as pd
import numpy as np
import json

import matplotlib

from sqlalchemy import create_engine
import psycopg2

import squarify


from flask import Flask, flash, redirect, render_template, request,   session, abort,send_from_directory,send_file,jsonify
import os


def connect_database():
    conn_string = 'postgresql+psycopg2://postgres:pass@127.0.0.1/reddit'
    db = create_engine(conn_string)

    conn = psycopg2.connect("dbname=reddit user=postgres password=Admin")
    conn.autocommit = True
    cursor = conn.cursor()
    return cursor

def execute_query_1(cursor):
    sql1='''select pixel_color, count(*) as color_count from pixel_placement where x < 10 and y < 10 group by pixel_color order by color_count desc;'
    cursor.execute(sql1)
    colors_result = pd.DataFrame(cursor.fetchall())
    return colors_result

def create_plot(colors_result):

    fig = px.bar(colors_result, x=0, y=1, color=colors_result[0], color_discrete_sequence=colors_result[0])
    fig = px.treemap(colors_result,names = ["A","B", "C", "D", "E"],
    parents = ["", "A", "B", "C", "A"])
    # data = [
    #     go.Bar(
    #         x=colors_result[0], # assign x as the dataframe column 'x'
    #         y=colors_result[1]
    #     )
    # ]

    graphJSON = json.dumps(fig, cls=plotly.utils.PlotlyJSONEncoder)

    return graphJSON
    # return plot
```

Page Elements



```
<div id="foo2"></div>
<script src="{{url_for('static', filename='js/plots_heat.js')}}"></script>
```

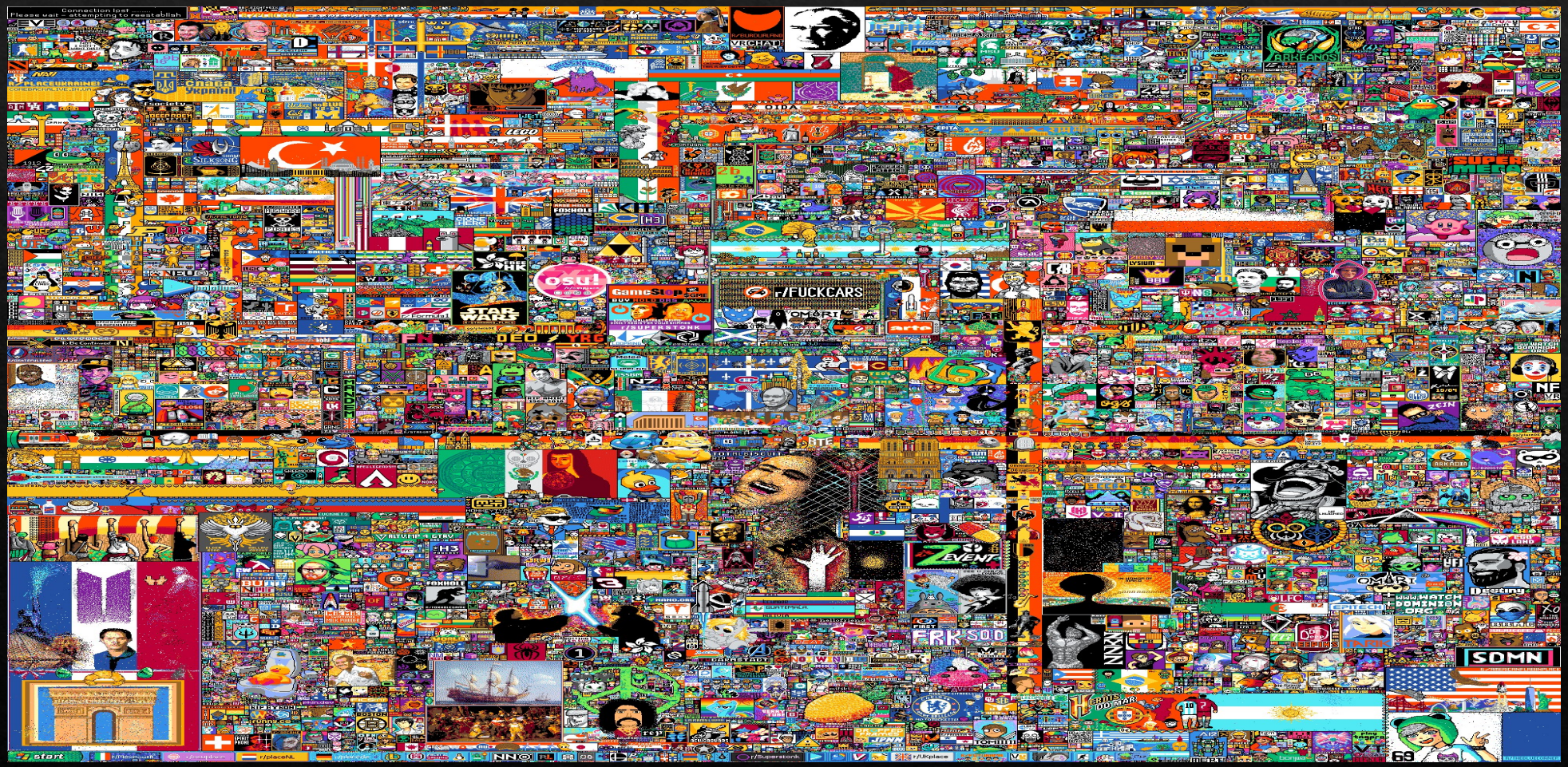
```
// Initializes the page with a default plot
function init() {
  data = [{
    z: heatmap_dataset.data,
    x: ['midnight', '1am', '2am', '3am', '4am', '5am', '6am', '7am', '8am', '9am', '10am', '11am', 'noon', '1pm', '2pm', '3pm', '4pm', '5pm', '6pm', '7pm', '8pm', '9pm', '10pm', '11pm'],
    y: heatmap_dataset.date_column,
    type: 'heatmap',
    hoverongaps: false
  }];
  layout = {
    xaxis: {
      title: 'Time of Day',
    },
    yaxis: {
      title: 'Date'
    }
  }
  Plotly.newPlot("foo2", data,layout);
}

// Call updatePlotly() when a change takes place to the DOM
d3.selectAll("#selDataset").on("change", updatePlotly);

// This function is called when a dropdown menu item is selected
function updatePlotly() {
  // Use D3 to select the dropdown menu
  var dropdownMenu = d3.select("#selDataset");
  // Assign the value of the dropdown menu option to a variable
  var dataset = dropdownMenu.property("value");

  // Initialize x and y arrays
  var x = [];
  var y = [];
  var z = [];

  if (dataset === 'dataset1') {
    z= heatmap_dataset.data,
    x= ['midnight', '1am', '2am', '3am', '4am', '5am', '6am', '7am', '8am', '9am', '10am', '11am', 'noon', '1pm', '2pm', '3pm', '4pm', '5pm', '6pm', '7pm', '8pm', '9pm', '10pm', '11pm'],
    y = heatmap_dataset.date_column,
    type = 'heatmap',
    hoverongaps = false
  }

  else if (dataset === 'dataset2') {
    z= heatmap_dataset2.data,
    x= ['midnight', '1am', '2am', '3am', '4am', '5am', '6am', '7am', '8am', '9am', '10am', '11am', 'noon', '1pm', '2pm', '3pm', '4pm', '5pm', '6pm', '7pm', '8pm', '9pm', '10pm', '11pm'],
    y= heatmap_dataset2.date_column,
    type= 'heatmap',
    hoverongaps= false
  }

  // Note the extra brackets around 'x' and 'y'
  Plotly.restyle("foo2", "x", [x]);
  // Plotly.restyle("foo2", "y", [y]);
  Plotly.restyle("foo2", "z", [z]);

}
init();
```

Where Clause

```
#DEFINING TWO DATASETS
bananada = [(176,494),(243,527)]
vhs =  [(1361,365),(1395,381)]
```

```
def generate_where_clause(top_left, bottom_right):
    return f'where x >= {top_left[0]} and x <= {bottom_right[0]} and y >= {top_left[1]} and y <= {bottom_right[1]}'

where_clause= generate_where_clause(
    bananada[0],bananada[1]
)

where_clause2 = generate_where_clause(
    vhs[0],vhs[1]
)
```

Final Canvas



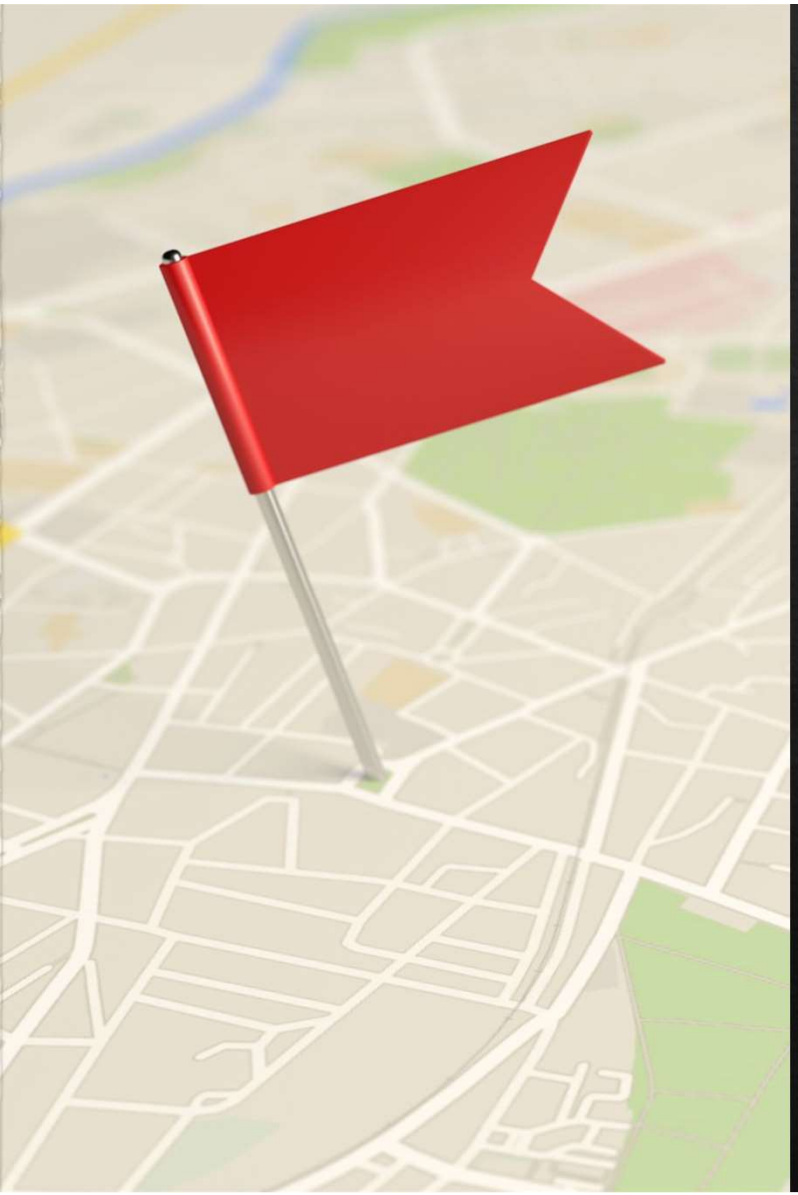
https://preview.redd.it/p3xgtndg7zr81.png?width=2000&format=png&auto=webp&s=00bc8eadb7b2f26e50bd065203c6a84ace70e29a

Final Comments

- Successes:
  - Data access solution
  - Where Clauses
- Challenges:
  - File size
  - Multiple moving pieces
- Lessons Learned:
  - r/place data archived permanently after a short time and the canvas wiped!

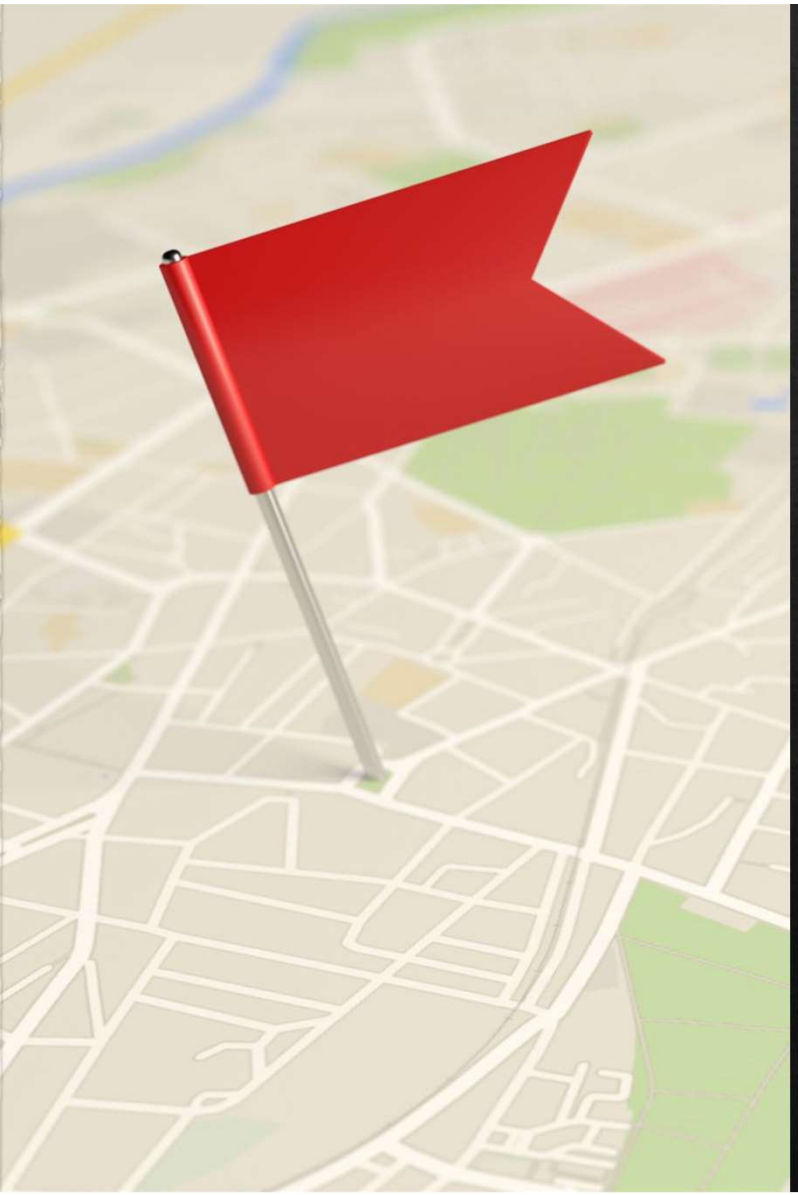
Data Resources

- https://www.reddit.com/r/place/comments/txvk2d/rplace_datasets_april_fools_2022/
- https://lospec.com/palette-list/r-place-2022-day3#:~:text=The%20selection%20of%20the%2032,Day%20event%20%2Fr%2Fplace.
- https://www.kaggle.com/datasets/robikscube/reddit-rplace-2022-history
- https://place.thatguyalex.com/
- https://place-atlas.stefanocoding.me/
- https://www.reddit.com/r/dataisbeautiful/search/?q=r%2Fplace&source=recent&restrict_sr=1
- https://www.reddit.com/r/place/comments/u1gy8t/all_20172022_rplace_projects/
- https://rplace.space/raw/



Coding References

- https://stackoverflow.com/questions/30610675/python-pandas-equivalent-in-javascript
- https://codepen.io/kurt_cagle/post/dataframes-come-to-javascript
- https://www.sqlservertutorial.net/sql-server-system-functions/convert-datetime-to-date/
- https://machinelearningknowledge.ai/matplotlib-heatmap-complete-tutorial-for-beginners/
- https://plotly.com/javascript/graphing-multiple-chart-types/
- https://plotly.com/python/linear-fits/



Coding References

- https://stackoverflow.com/questions/33282368/plotting-a-2d-heatmap-with-matplotlib
- https://dev.to/gustavosfq/convert-an-image-to-html-pixel-by-pixel-3m1n
- https://www.w3schools.com/tags/canvas_drawimage.asp
- https://d3js.org/
- https://docs.sqlalchemy.org/en/14/orm/declarative_tables.html
- https://towardsdatascience.com/combining-python-and-d3-js-to-create-dynamic-visualization-applications-73c87a494396
- https://www.geeksforgeeks.org/treemap-using-plotly-in-python/