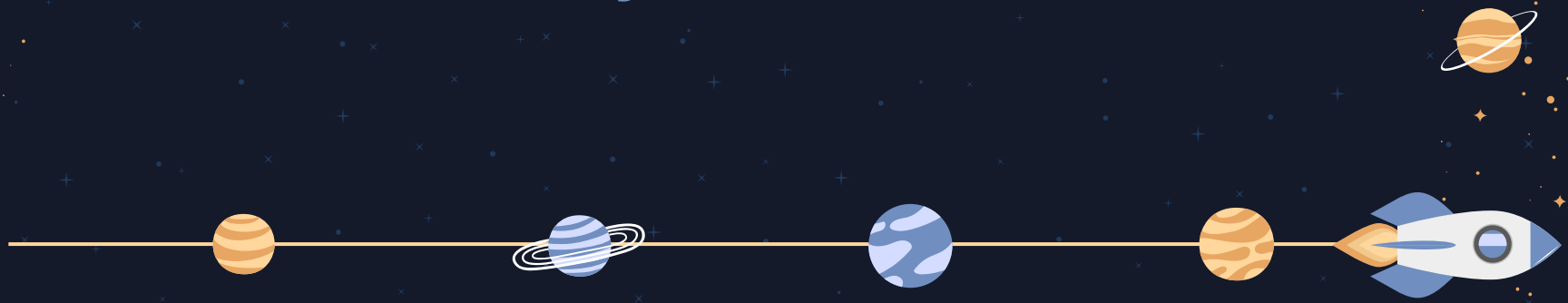
Project 3 – Visualizing Data

Team Members: Jananee Arjunan, Samuel Heaton, Mia Tsivitse, Alice Johnson, AJ Domingo

Project Overview



Section .01

Project Scope

Section .02

Data | ETL

Section .03

Back End

Section .04

Front End

01

Project Scope

2,060

Reported UFO sightings in the U.S. from July - December 2022

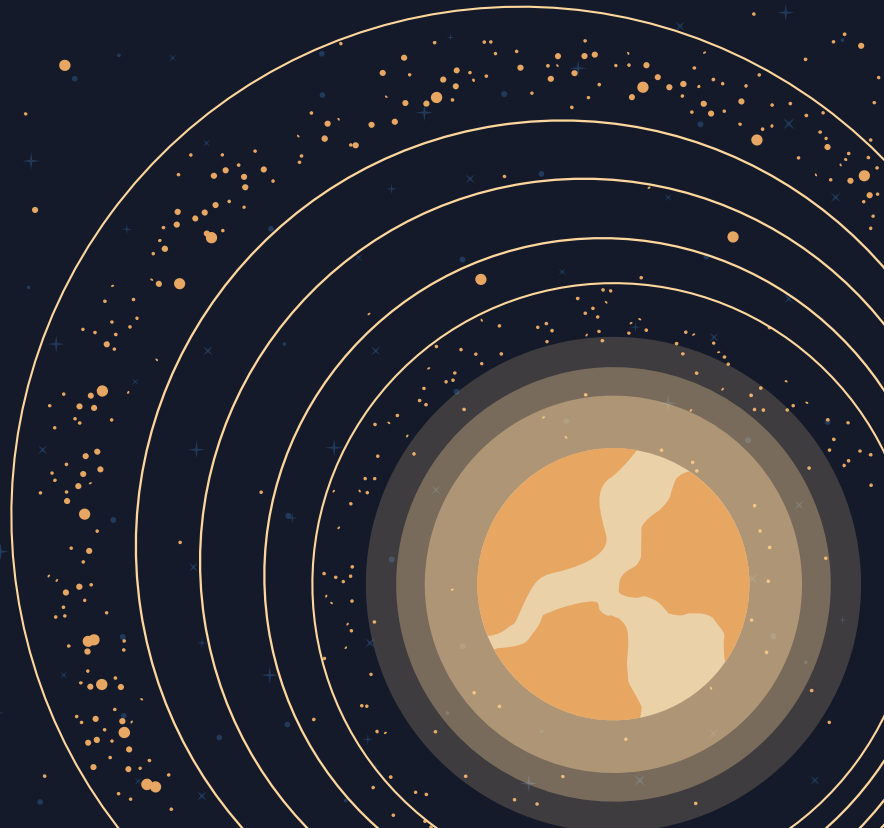
Source: NUFORC.org

Project Scope

UFO sightings have been reported in all 50 U.S. states during the last century, and the sightings have varied greatly in terms of overall characteristics such as the UFO shape and size.

We explored data from the National UFO Reporting Center (NUFORC) on UFO sightings within the U.S. during July - December 2022. During 2022, there were **4.3K** total UFO sightings and **2K** occurred in recent months (July - December 2022).

We discovered patterns in the sightings reported to the NUFORC and showcased data through visualizations such as a Leaflet heatmap and Plotly charts.



02

Data

Data Research + ETL

Data Research

- NUFORC.org (National UFO Reporting Center) has collected data from 150,000 self-reported UFO sightings over 47 years

Data Transformation

- Use data gathered from NUFORC.org & Google Maps API to sort UFO sightings data and create filtered data sets specific to month of year, hour of day and U.S. State.

Data Extraction

- Scrape UFO sightings data from NUFORC.org from July - December 2022 from all U.S. States
- Pull coordinates from Google Maps API to support our visualizations

Data Load

- Create four tables in Postgresql under ufo_sightings database
  - All Sightings
  - Sightings by Month
  - Sightings by Hour of Day
  - Sightings by State

Data Extraction

In [8]:
```
# Extract all data
table = b_soup.find('font', style_='FONT-SIZE:11pt')
records = b_soup.find_all('tr', valign='TOP')
```

In [9]:
```
# New empty list
sep_nuforc_df= []
# Loop through data, create a list of rows
for record in records:
    record_info = {}
    date = record.find_all('td')[0].text
    city = record.find_all('td')[1].text
    state = record.find_all('td')[2].text
    country = record.find_all('td')[3].text
    shape = record.find_all('td')[4].text
    duration = record.find_all('td')[5].text
    summary = record.find_all('td')[6].text
    posted = record.find_all('td')[7].text
    images = record.find_all('td')[8].text
#     print fields

    record_info = {
    "Date": date,
    "City" : city,
    "State" : state,
    "Country": country,
    "Shape": shape,
    "Duration" : duration,
    "Summary" : summary,
    "Posted" : posted,
    "Images" : images
    }

    sep_nuforc_df.append(record_info)
record_info
```

Out[9]:
```
{'Date': '9/1/22 01:23',
 'City': 'Chesapeake',
 'State': '',
 'Country': 'USA',
 'Shape': 'Delta',
 'Duration': '',
 'Summary': 'I was in my room and I seen three entities
```

In [11]:
```
# loop through the df
for index,row in city_df.iterrows():
    city = row["Location"]
    url = "https://maps.googleapis.com/maps/api/geocode/json?address="
    print(f"Processing Record for {city}")
    #     get the response
    # Use try and except to skip the missing data
    try:
        query_url = f"{url}{city}&key={API_key}"
        response = requests.get(query_url).json()
#         pprint(response)
        city_df.loc[index,"Latitude"] = response['results'][0]['geometry']['location']['lat']
        city_df.loc[index,"Longitude"]= response['results'][0]['geometry']['location']['lng']
    except(KeyError, IndexError, JSONDecodeError):
        print(f"City not found. Skipping...")
    except requests.ConnectionError:
        print("ConnectionError...")
```

```
Processing Record for West Lafayette, OH

C:\Users\13135\anaconda3\envs\PythonData\lib\site-packages\pandas\core\indexing.py:1765: Setti
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/
versus-a-copy
  isetter(loc, value)

Processing Record for Burr Ridge, IL
Processing Record for Conway, SC
Processing Record for Keyser, WV
Processing Record for gressham, OR
Processing Record for Eddyville, KY
Processing Record for Carpinteria, CA
Processing Record for Tacoma, WA
Processing Record for Charlotte, NC
Processing Record for Burnet, TX
Processing Record for Seattle, WA
```

Above: Scraping Data from NUFORC
Above Right: API Calls to Google Maps API

Data Transformation

```
In [10]: # Create DataFrame
sep_data_df=pd.DataFrame(sep_nuforc_df)
sep_data_df
```

Out[10]:

	Date	City	State	Country
0	9/30/22 22:51	Stevens Point	WI	USA
1	9/30/22 21:50	Lawrenceville	GA	USA
2	9/30/22 21:00	Grand junction	CO	USA
3	9/30/22 20:48	West valley city	UT	USA
4	9/30/22 20:36	Flora	IL	USA
...	...	...	...	...
511	9/1/22 09:14		MT	USA
512	9/1/22 08:00	Nazareth	PA	USA
513	9/1/22 02:43	Hilton	NY	USA
514	9/1/22 01:30	Fayetteville	NC	USA
515	9/1/22 01:23	Chesapeake		USA

516 rows × 9 columns

```
In [12]: city_df.head(150)
reduced_df = city_df[["ReportNum","Location","Latitude","Longitude"]]
reduced_df
```

Out[12]:

	ReportNum	Location	Latitude	Longitude
0	1	West Lafayette, OH	40.2753	-81.751
1	2	Burr Ridge, IL	41.7489	-87.9184
2	3	Conway, SC	33.836	-79.0478
3	4	Keyser, WV	39.4409	-78.9739
4	5	gressham, OR	45.5045	-122.436
...	...	...	...	...
2126	2127	Snowflake, AZ	34.5134	-110.078
2127	2128			
2128	2129	El paso, TX	31.7619	-106.485
2129	2130	CHARLESTOWN, IN	38.4531	-85.6702
2130	2131	Mansfield, TX	32.5632	-97.1417

```
In [30]: #Summary of UFO Sightings by State
print("Count of UFO Sightings by State:")
states = ufo_data.groupby(['State']).size().to_frame(name = 'Sightings').reset_index()
states
```

Count of UFO Sightings by State:

Out[30]:

	State	Sightings
0	AK	3
1	AL	28
2	AR	18
3	AZ	64
4	CA	203
5	CO	58
6	CT	35
7	DC	3
8	DE	11
9	FL	152
10	GA	35
11	HI	5
12	IA	14
13	ID	24
14	IL	58
15	IN	38
16	KS	12
17	KY	27
18	LA	21
19	MA	53
20	MD	31

```
In [9]: #Resample Months
ufo_months = ufo_data.set_index('Date').resample('M').count()
ufo_months = ufo_months.reset_index()
ufo_months
```

Out[9]:

	Date	City	State	Country	Shape	Duration	Summary	Posted	Images	ReportNum	Location	Latitude	Longitude
0	2022-07-31	399	399	399	382	379	399	399	114	399	399	397	397
1	2022-08-31	419	419	419	409	408	419	419	126	419	419	418	418
2	2022-09-30	455	455	455	455	444	455	455	134	455	455	455	455
3	2022-10-31	393	393	393	391	375	393	393	115	393	393	392	392
4	2022-11-30	264	264	264	264	253	264	264	74	264	264	264	264
5	2022-12-31	129	129	129	129	120	129	129	29	129	129	127	127

```
In [10]: #Summary of UFO Sightings by Month
print("Count of UFO Sightings by Month:")
ufo_monthly = ufo_months.groupby('Date').sum()['State'].to_frame(name = 'Sightings').reset_index()
ufo_monthly
```

Above Left: Organize NUFORC Data into DataFrames
Above Middle: Combine Coordinate Data with NUFORC Data
Above: Groupby NUFORC Data for Sightings by State
Left: Groupby NUFORC Data for Sightings by Month

Data Load

ufo_sightings/postgres@PostgreSQL 15

No limit

Query | Query History

```sql
DROP TABLE if exists sighting;

CREATE TABLE sighting (
Date TIMESTAMP,
City TEXT,
State TEXT,
Country TEXT,
Shape TEXT,
Duration varchar,
Summary varchar,
Posted date,
Images varchar,
ReportNum int Primary key,
Location varchar,
Latitude float,
Longitude float);

select * from sighting;
```

Tables (4)
- hours
- months
- sighting
- states

```
In [1]:
import pandas as pd
from sqlalchemy import create_engine
from sqlalchemy import inspect
from config import db_url

In [2]:
# Create engine to connect database
engine = create_engine(f'{db_url}')
connection = engine.connect()

In [3]:
inspector = inspect(engine)

inspector.get_table_names()

Out[3]: ['hours', 'sighting', 'states', 'months']

In [4]:
sighting = pd.read_sql("SELECT * FROM sighting",connection)
sighting
```

Out[4]:

	date	city	state	country	shape	duration	summary
0	2022-07-31 21:59:00	West Lafayette	OH	USA	Light	4 minutes	Bright circle light moving through sky.
1	2022-07-31 21:45:00	Burr Ridge	IL	USA	Other	None	Strange red, blue and black craft in the sky a...
2	2022-07-31 21:15:00	Conway	SC	USA	Circle	Less than a minute	Round bright white light go in a straight line...
	2022-						Saw an

Table #1 - Sighting

All UFO Sightings

Table #2 - States

Sightings by U.S. State

Table #3 - Months

Sightings by Month

Table #4 - Hours

Sightings by Hour of Day (UTC)

03

Back End

Back End

app.py > ...

```python
import sqlalchemy
from sqlalchemy.ext.automap import automap_base
from sqlalchemy.orm import Session
from sqlalchemy import create_engine,func
from flask import Flask,jsonify, render_template
import pandas as pd
from config import db_url

#################################################
# Database Setup
#################################################
engine = create_engine(f'{db_url}')
connection = engine.connect()

# reflect an existing database into a new model
Base = automap_base()
# reflect the tables
Base.prepare(engine,reflect=True)

# Save reference to the table
Sighting = Base.classes.sighting
State = Base.classes.states
Month = Base.classes.months
Hour = Base.classes.hours
```

```python
#################################################
# Flask Routes
#################################################
# Home route
@app.route("/")
# def Welcome():
#     return(
#         f"<b>Welcome</b>"
#     )
def index():
    return render_template("index.html", pages={
        "Home": "active",
        "Sightings_Map": "",
        "Sightings_Trends": "",
        "UFO_Shapes": ""
    })

@app.route("/Sightings_Map")
def sightings_map():
    return render_template("Sightings_Map.html", pages={
        "Home": "",
        "Sightings_Map": "active",
        "Sightings_Trends": "",
        "UFO_Shapes": ""
    })
```

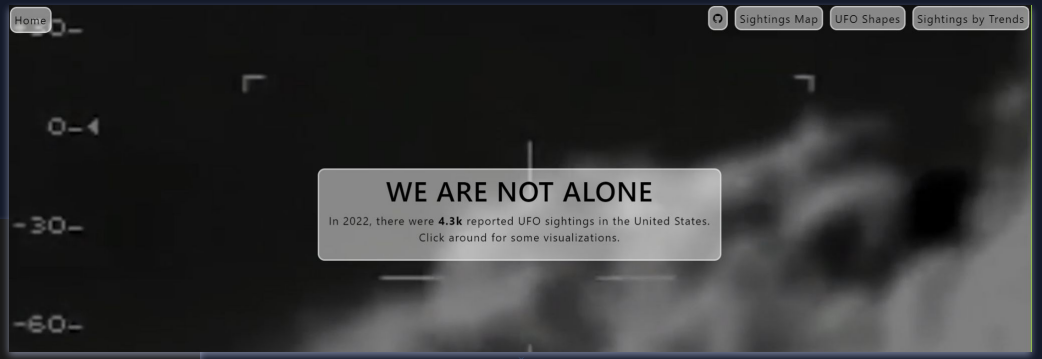
```
Data
static
  css
  img
  js
templates
.gitignore
app.py
config.py
```

```python
@app.route("/api/v1.0/sightingsbymonth/")
def sightings_month():
    # Create our session (link) from Python to the DB
    session = Session(engine)

    month_data = session.query(Month.id,Month.date,Month.sightings).all()

    session.close()

    month_list=[]
    for data in month_data:
        month_dict={}
        month_dict["Date/Time"]=data.date.strftime("%B")
        month_dict["sightings"]=data.sightings
        month_list.append(month_dict)

    return jsonify(month_list)

@app.route("/api/v1.0/sightingsbyhour/")
def sightings_hour():
    # Create our session (link) from Python to the DB
    session = Session(engine)

    hour_data = session.query(Hour.id,Hour.hour,Hour.sightings).all()

    session.close()

    hour_list=[]
    for data in hour_data:
        hour_dict={}
        hour_dict["hour"]=data.hour
        hour_dict["sightings"]=data.sightings
        hour_list.append(hour_dict)

    return jsonify(hour_list)


# app.run statement
if __name__ == '__main__':
    app.run(debug=True)
```

Above Left: Connect to Database (Postgresql)
Above Middle: Flask Routes
Above Right: Flask Routes
Right: File Organization

Pages Rendered by Flask

Right: Index.html
Below Right: State and Hour JSON Response
Below Left: Sightings JSON Response



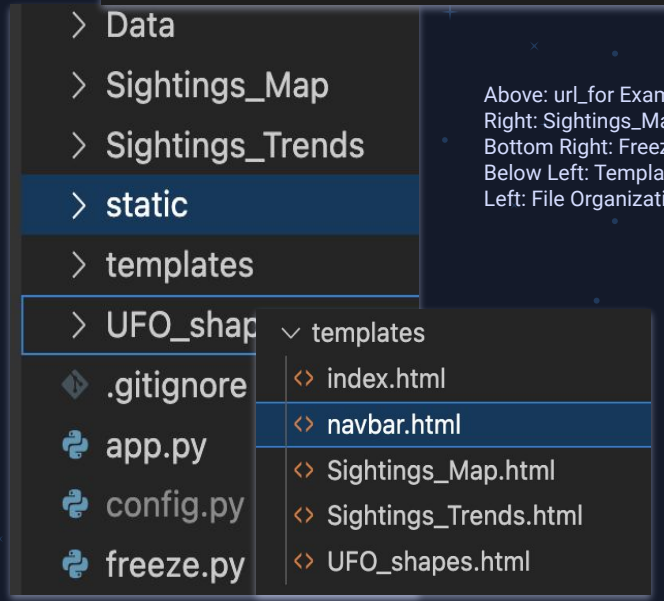

```
[
  {
    "City": "West Lafayette",
    "Country": "USA",
    "Date": "Sun, 31 Jul 2022 21:59:00 GMT",
    "Duration": "4 minutes",
    "Images": "Yes",
    "Location": "West Lafayette, OH",
    "Posted Date": "Fri, 09 Sep 2022 00:00:00 GMT",
    "Report Num": 1,
    "Shape": "Light",
    "State": "OH",
    "Summary": "Bright circle light moving through sky.",
    "coordinates": [
      "40.2753487",
      "-81.7509601"
    ]
  },
  {
    "City": "Burr Ridge",
    "Country": "USA",
    "Date": "Sun, 31 Jul 2022 21:45:00 GMT",
    "Duration": null,
    "Images": "Yes",
    "Location": "Burr Ridge, IL",
    "Posted Date": "Fri, 09 Sep 2022 00:00:00 GMT",
    "Report Num": 2,
    "Shape": "Other",
    "State": "IL",
    "Summary": "Strange red, blue and black craft in the sky and green light below it just above the tree line",
    "coordinates": [
      "41.74892029999999",
      "-87.91839189999999"
    ]
  },
```

```
[
  {
    "sightings": 3,
    "state": "AK"
  },
  {
    "sightings": 28,
    "state": "AL"
  },
  {
    "sightings": 18,
    "state": "AR"
  },
  {
    "sightings": 64,
    "state": "AZ"
  },
```

```
[
  {
    "hour": 0,
    "sightings": 84236
  },
  {
    "hour": 1,
    "sightings": 74363
  },
  {
    "hour": 2,
    "sightings": 69322
  },
  {
    "hour": 3,
    "sightings": 66517
  },
  {
    "hour": 4,
    "sightings": 56597
  },
```

Frozen Flask / Github Pages Deployment

```
<!-- Our own CSS stylesheet -->
<link rel="stylesheet" href="{{ url_for('static', filename='css/styles.css') }}"
```



Above: url_for Example for Calling CSS Stylesheet
Right: Sightings_Map.html Example
Bottom Right: Freezer.py Code
Below Left: Templates Folder
Left: File Organization for Flask Freezer

ates > Sightings_Map.html > ...

```
{% extends 'navbar.html' %}


{% block title %}Sightings Map{% endblock %}


{% block head %}
<!-- Leaflet JS code-->
<script src="https://unpkg.com/leaflet@1.6.0/dist/leaflet.js"
  integrity="sha512-gZwIG9x3wUXg2hdXF6+rVkLF/0Vi9U8D2Ntg4Ga5I5BZpV
  crossorigin=""></script>
```

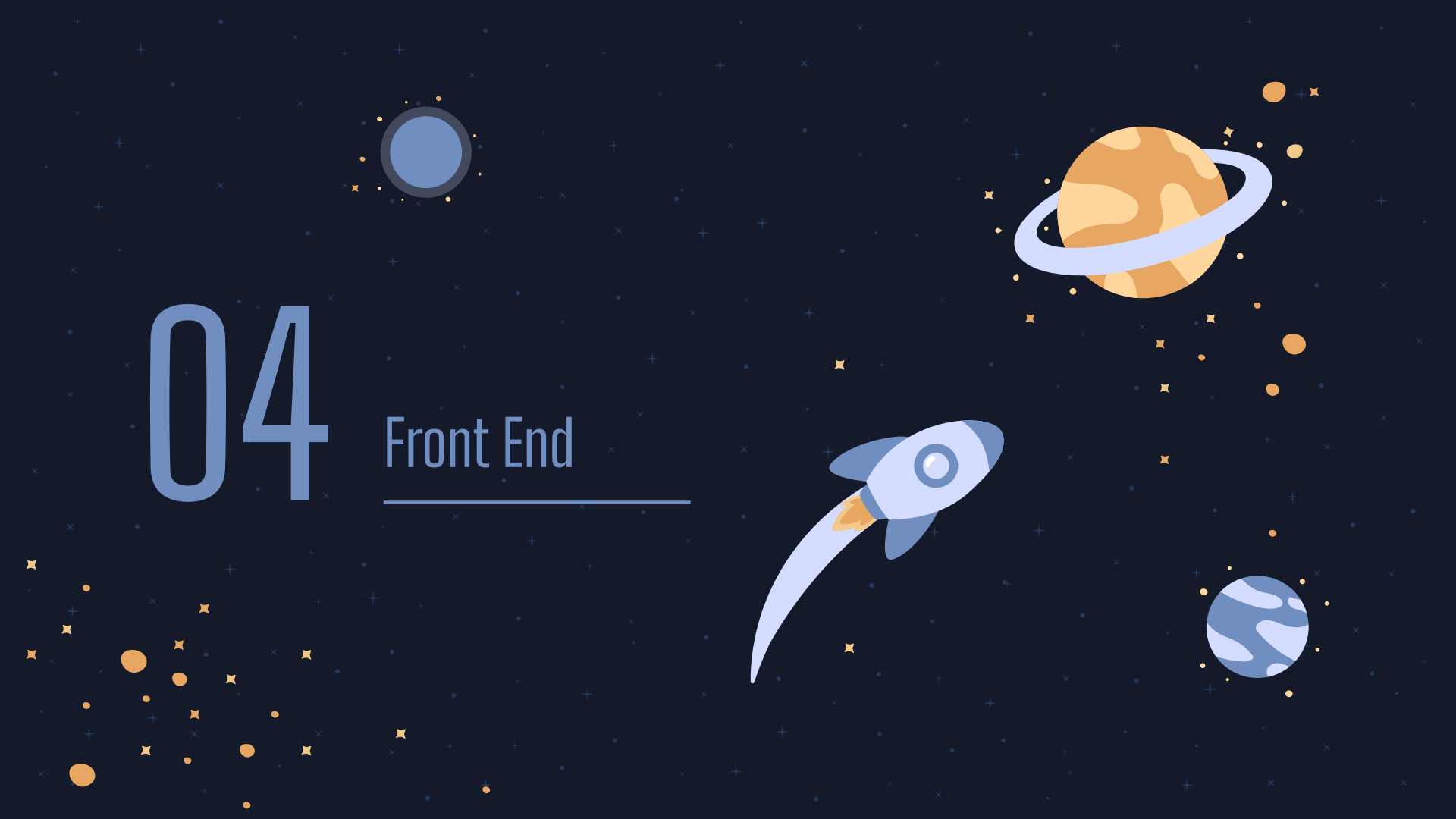
freeze.py 1

freeze.py > ...

```
1  from flask_frozen import Freezer
2  from app import app
3  from config import base_url
4
5  if __name__ == '__main__':
6      app.config["FREEZER_BASE_URL"] = base_url
7      freezer = Freezer(app)
8      freezer.freeze()
```

```
src="{{ url_for('static', filename=

d3.v5.min.js"></script>

ttps://unpkg.com/leaflet@1.6.0/dist/
BIphAcBb3F6JVqxf46+CDLwfLMHloNu6KEQC

('sightings')}}";</script>
```

04 Front End



Front End

bideo.js

```
static > js > plugin_bideo > bideo.js > ...
/**
 * Full Background Video
 *
 * More info on Audio/Video Media Events/Attributes/Methods
 * - https://developer.mozilla.org/en-US/docs/Web/Guide/Events/Media_events
 * - http://www.w3schools.com/tags/ref_av_dom.asp
 */

(function (global) {

  // Define Bideo constructor on the glo
  global.Bideo = function () {

    // Plugin options
    this.opt = null;
    // The Video element
    this.videoEl = null;
```

navbar.html

```
templates > navbar.html > ...
<!doctype html>

<head>
    <meta charset="utf-8">
    <meta content="width=device-width, initial-scale=1" name="viewport">
    <title>US UFO Sightings - {% block title %}{% endblock %}</title>

    <!-- JQuery -->
    <script src="https://code.jquery.com/jquery-1.11.3.min.js"></script>
    <meta charset="utf-8">
    meta content="width=device-width, initial-scale=1" name="viewport">
    meta http-equiv="X-UA-Compatible" content="IE=edge">

!-- reset styling -->
link rel="stylesheet" href="{{ url_for('static', filename='css/reset.css') }}">
```

visualization_bg.js

```
static > js > visualization_bg.js > granimInstance
let granimInstance = new Granim({
  element: '#canvas-image-blending',
  direction: 'top-bottom',
  isPausedWhenNotInView: true,
  image: {
    source: "/Group-Project-3/static/img/
    bg-forest-fullscreen.jpg",
    blendingMode: 'multiply',
    stretchMode: ['stretch', 'stretch'],
  },
  states: {
    "default-state": {
      gradients: [
        ['#29323c', '#485563'],
        ['#FF6B6B', '#556270'],
        ['#80d3fe', '#7ea0c4'],
        ['#f0ab51', '#eceba3']
      ],
      transitionSpeed: 9000
    }
  }
});
```

styles.css

```
static > css > # styles.css > ...
/* nav elements */
.navbar {
  background-color: transparent;
  padding-inline: 10px;
}

/* plugin_bideo */
html,
body {
  width: 100%;
  height: 100%;
  overflow: hidden;
}
```

Above: Full Background Video (Bideo JS Library)
Above Right: Navigation Bar
Below Right: CSS Styling
Middle Right: Background Image Transitions

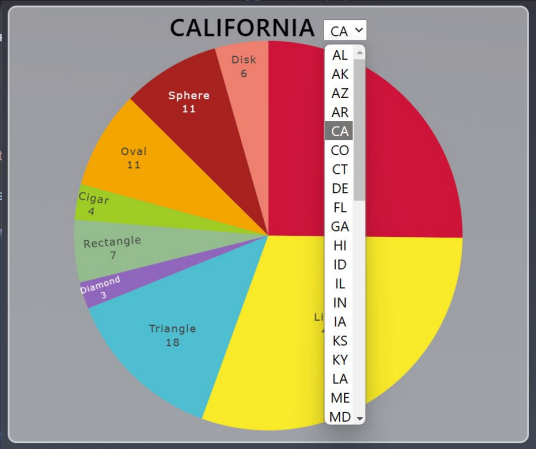
Visualizations – UFO Sightings by Shape

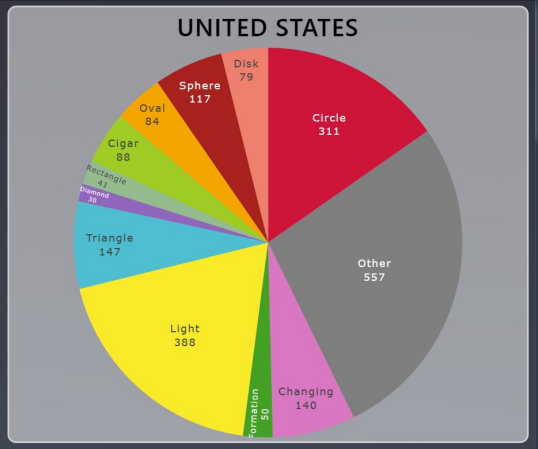
UFO_shapes.js

```
static > js > UFO_shapes.js > ...
16
17  // Make empty arrays the same length as common_shapes_labels
18  state_shapes_values = [];
19  us_shapes_values = [];
20  for (let i = 0; i < common_shapes_labels.length; i++) {
21      state_shapes_values.push(0);
22      us_shapes_values.push(0);
23  };
24
25  // Function to sort sighting shapes according to common_shapes_labels
26  function filterShapes(labels, values, shape) {
27      if (labels.includes(shape)) {
28          values[labels.indexOf(shape)]++;
29      } else {
30          values[labels.indexOf('Other')]++
31      };
32  };
33
34  function init() {
35      // Dropdown to choose any US State
36      let dropDown = d3.select('#selectedSt
37      for (let i = 0; i < usStates.length;
38          dropDown.append("option").text(us
39      };
```

UFO_shapes.html

```
templates > UFO_shapes.html > ...
20  {% block content %}
21  <!-- Start of Visualization -->
22  <div class="container-fluid">
23    <div class="row justify-content-center">
24      <div class="col-md-5 content-background">
25        <!-- Start of State Pie Chart -->
26        <h3 id="selectedStateName">ALABAMA </h3><select id="selectedStateAbbrev"
27          onchange="optionChanged(this.value)"></select>
28        <div id="statePieChart"></div>
29        <!-- End of State Pie Chart -->
```





Visualizations - Leaflet Heatmap

leaflet-heatmap.js

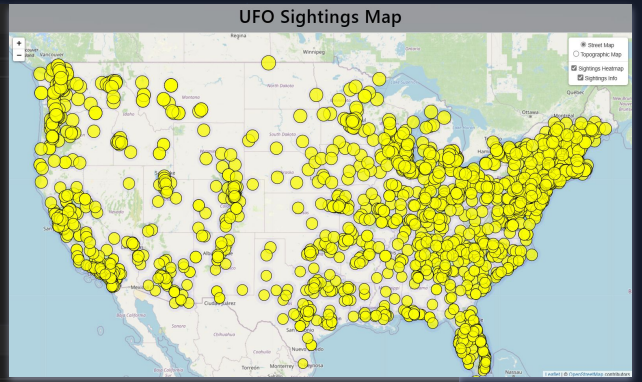
static > js > leaflet-heatmap.js > ...

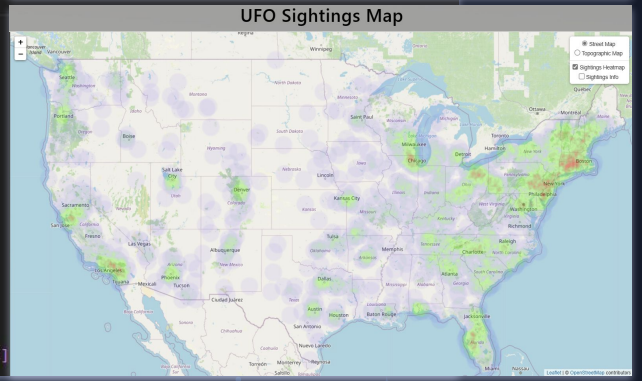
```
/*
 * Leaflet Heatmap Overlay
 *
 * Copyright (c) 2008-2016, Patrick Wied (https://www.patrick-wied.at)
 * Dual-licensed under the MIT (http://www.opensource.org/licenses/mit-license.php)
 * and the Beerware (http://en.wikipedia.org/wiki/Beerware) license.
 */
;(function (name, context, factory) {
  // Supports UMD. AMD, CommonJS/Node.js and browser context
  if (typeof module !== "undefined" && module.exports) {
    module.exports = factory(
      require('heatmap.js'),
      require('leaflet')
```

Sightings_Map.js

static > js > Sightings_Map.js > ...

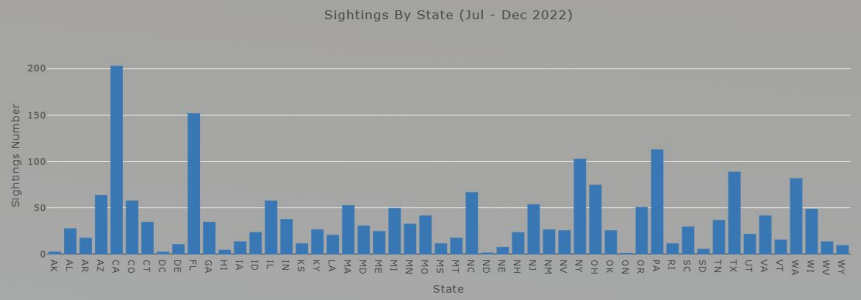
```
// // leaflet
d3.json(url).then(function (response) {
  // console.log(response);

  markers = [];
  heatArray = [];
  for (var i = 0; i < response.length; i++) {
    // console.log(response[i].coordinates);
    if (!response[i].coordinates.includes(null)) {
      // if (response[i].coordinates[0] !== null && response[i].coordinates[1] !== null){


      markers.push(L.circle(response[i].coordinates, {
        fillOpacity: 0.75,
        color: "black",
        fillColor: "yellow",
        radius: 50000,
        weight: 1
      }).bindPopup(`<h3>${response[i].Location}</h3><hr><b>Date Reported :</b> ${response[i]
```



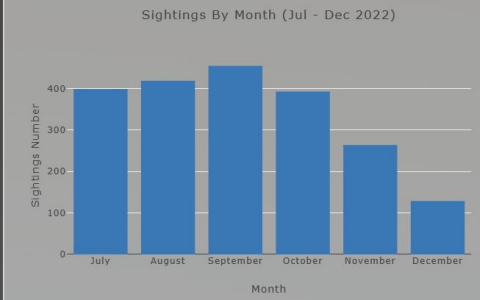


Visualizations – Sightings Trends

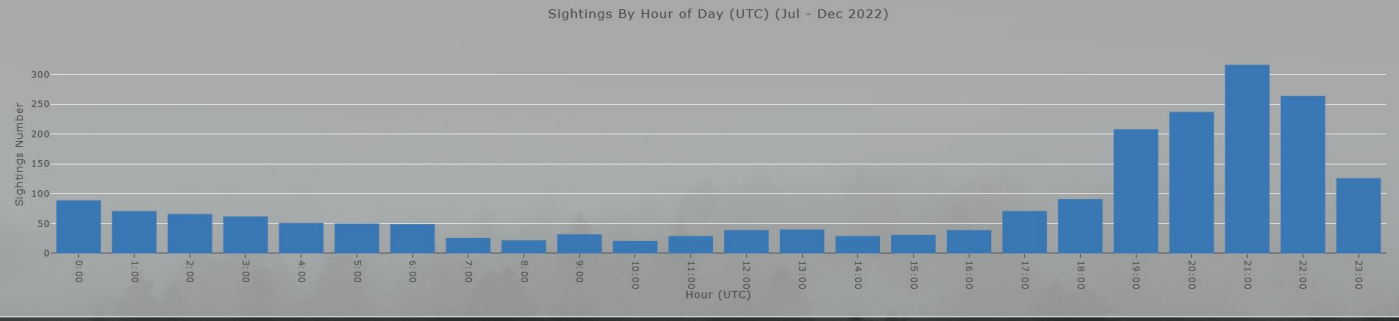
Sightings by State



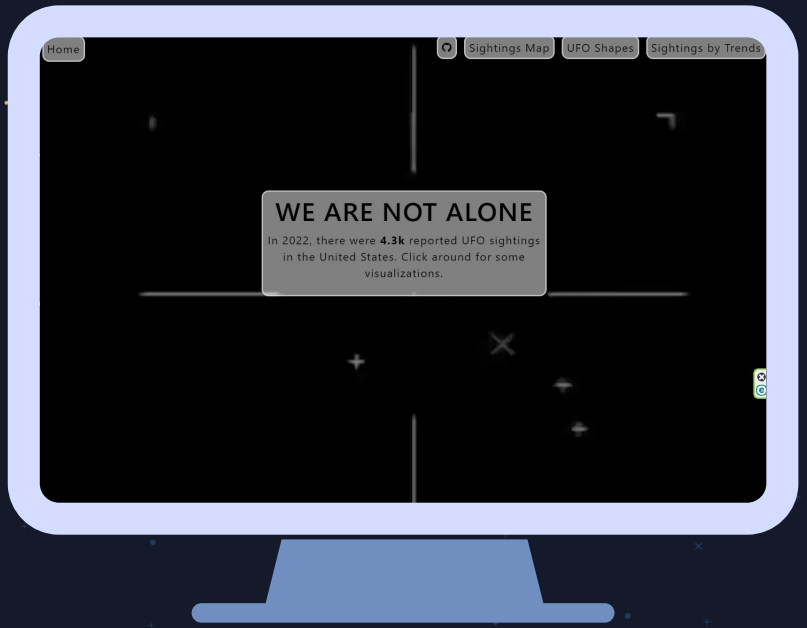
Sightings by Month



Sightings by Hour of Day (UTC)



Our Dashboard



Project Link

Thank You

Questions?