Web Visualizations of 2016 and 2020 United States Presidential Elections

Group 1
Ben Stork
Leah Brady
Cristina Sheridan
Jermaine Coleman

Important Questions

We had three questions we wanted to try and answer.

1. How did counties vote in the 2016 Presidential Election?
2. How did counties vote in the 2020 Presidential Election?
3. What was the increase in total voters in each county from 2016 to 2020?

There was an interest in seeing how complicated it was to have an interactive map that showed how counties voted for both 2016 and 2020 since these were put together as a simple way to convey complicated information at a variety of news sites. This project would give the group members insight into how complicated it is to become a data analyst.

Our Data Sources

The Following Data Sources Were Utilized:

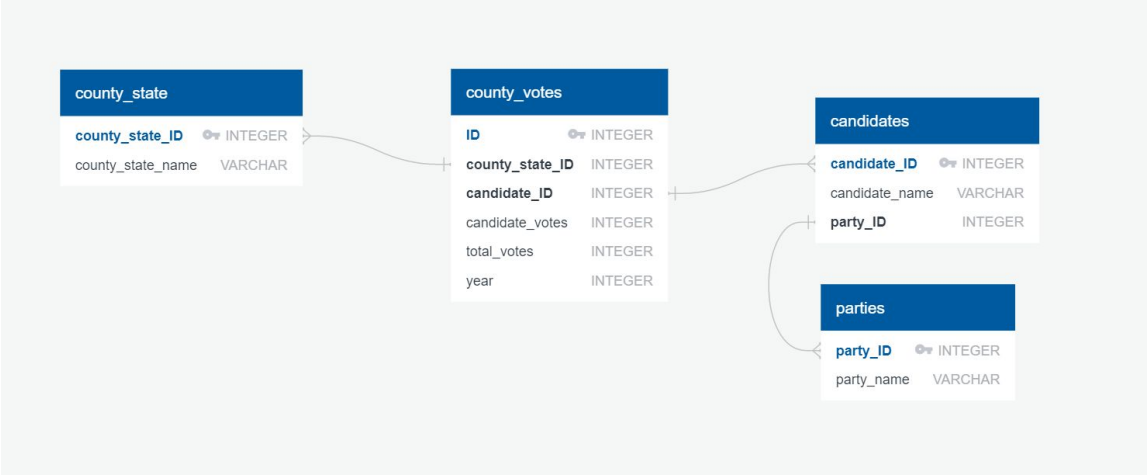
2016 Election Data - from MIT Election Data + Science Lab (secondary source that had files converted to CSV's)
https://github.com/MEDSL/county-returns

2020 Election Data- (we believe data was obtained from the BBC, the following is linked https://www.bbc.com/news/election-us-2020-53785985)
https://www.kaggle.com/unanimad/us-election-2020?select=president_county_candidate.csv

Geojson Data for Counties and States - (United States Census Cartographic Boundary Files converted to GeoJSON files using the MyGeoData vector converter)
https://eric.clst.org/tech/usgeojson/

Create ERD and SQL Schema (Leah)

Created the visualization and the structure of the database, which would provide the underlying foundation for accessing the data.



Data Exploration and Clean Up Process (Ben)

First all Data Sets needed to be imported

1) Three different CSV files need to be imported and combined into a single dataset
2) Original dataframes were the following size
   a) 50524 x 11
   b) 31148 x 6
   c) 4633 x 5
3) Final dataframe was the following size after all cleaning and combining
   a) 15301 x 7

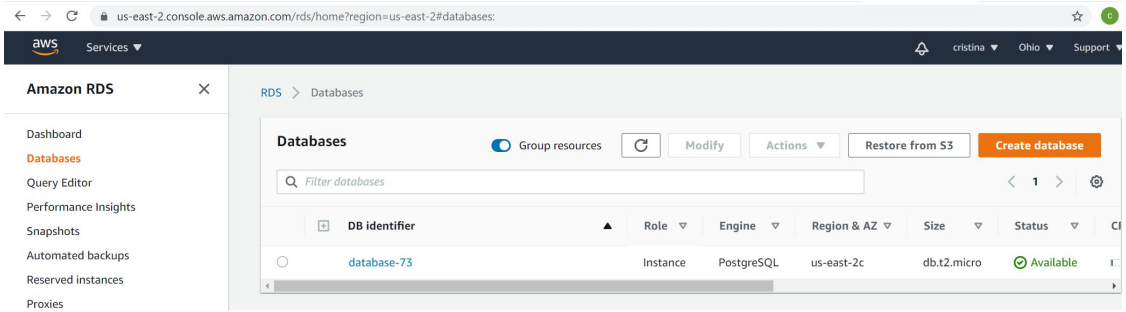
```python
#Reading in Elections CSV files
election_2016_path = "data/countypres_2000-2016.csv"
election_2016_df = pd.read_csv(election_2016_path)

election_2020_path= "data/pres20results.csv"
election_2020_df = pd.read_csv(election_2020_path)

election_2020_total_votes= "data/president_county20.csv"
election_2020_total_votes_df = pd.read_csv(election_2020_total_votes)
```

election_2016_df	DataFrame	(50524, 11)	year state state_po county FIPS office \ 0 2000 Ala <..
election_2020_df	DataFrame	(31148, 6)	state county candidate party total_votes won 0 Delaware
election_2020_total_votes_d	DataFrame	(4633, 5)	state county current_votes total_votes \ 0 <...> 99 462

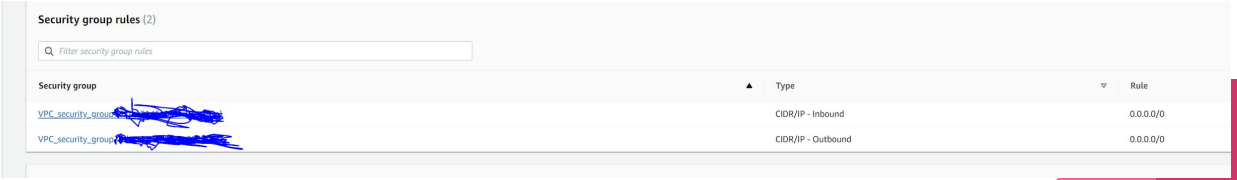
election_merged_df	DataFrame	(15301, 7)	ID year candidate party candidatevotes totalvotes count

Database server hosted on AWS (Cristina)

The group was able to get the databases hosted on AWS but had issues being able to get the API pushed up to AWS due to unfamiliarity and steep learning curve.



Database Hosted on AWS Server



Security Group Rules had to be Altered for Member Access

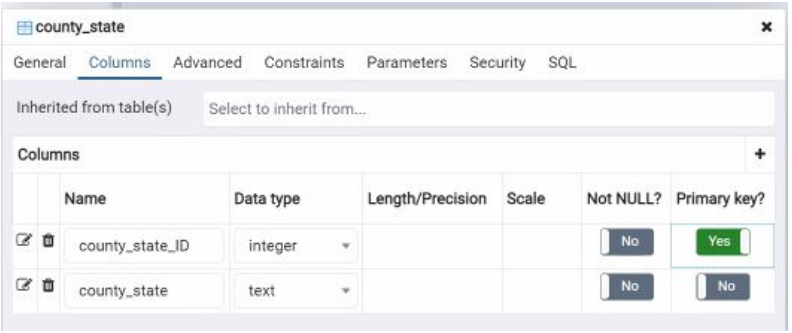
SQL Data Upload and Postgres Setup (Cristina)

1) All tables needed to be pushed up to Postgres with the following code

```
#take tables to #take tables to postgress to election_db database
rds_connection_string = "postgres:fpEl1620@database-73.cgiopwhezyby.us-east-2.rds.amazonaws.com/Election_db"
engine = create_engine(f'postgresql://{rds_connection_string}')
engine.table_names()
county_state_df.to_sql(name='county_state', con=engine, if_exists='replace', index=False)
county_votes_df.to_sql(name='county_votes', con=engine, if_exists='replace', index=False)
parties.to_sql(name='parties', con=engine, if_exists='replace', index=False)
candidates.to_sql(name='candidates', con=engine, if_exists='replace', index=False)
```

2) When uploading to Postgres we did not import schema first and then ran into issues getting the API pulled down due to lack of Primary Key.

3) After Primary Keys were added as shown to all tables we were able to get the API working.



API completed with app.py on Local Server (Ben)

1) App.py was completed based on class activity 10.3.10. Without Primary Keys app.py was failing on the table setup prior to flask setup.

2) All postgres tables were pulled and defined in the api in a similar manner.

3) API was able to be referenced from a local host but was not implemented with AWS or Heroku.

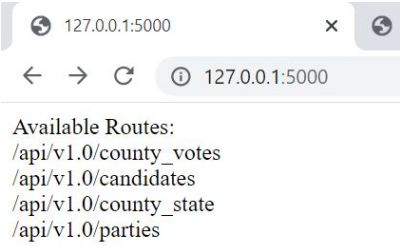
```
# Save reference to the tables
County_votes = Base.classes.county_votes
Candidates = Base.classes.candidates
County_state = Base.classes.county_state
Parties = Base.classes.parties
```

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

    # Query all county_votes info
    results = session.query(County_votes.ID, County_votes.year, County_votes.candidate_ID, Count

    session.close()

    # Create a dictionary from the row data and append to a list of county_votes_list
    county_votes_list = []
    for ID, year, candidate_ID, candidate_votes, total_votes, county_state_ID in results:
        county_votes_dict = {}
        county_votes_dict['ID']= ID
        county_votes_dict['year']= year
        county_votes_dict['candidate_ID']= candidate_ID
        county_votes_dict['candidate_votes']= candidate_votes
        county_votes_dict['total_votes']= total_votes
        county_votes_dict['county_state_ID']= county_state_ID
        county_votes_list.append(county_votes_dict)
    return(jsonify(county_votes_list))
```

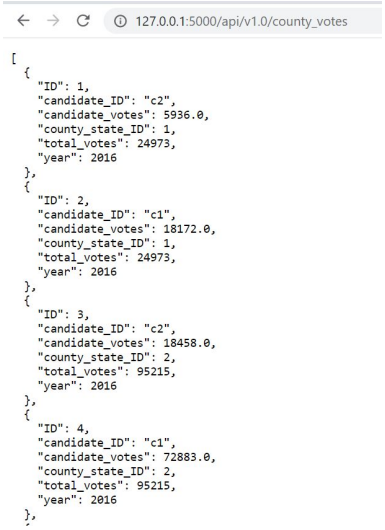
API Running on Local Host (Ben)

1) Once all issues with Primary Keys were resolved the api was able to be launched in the local browser.

2) All four postgres tables were able to be pulled in including: county_votes, county_state, candidates, and parties



Main Menu of API

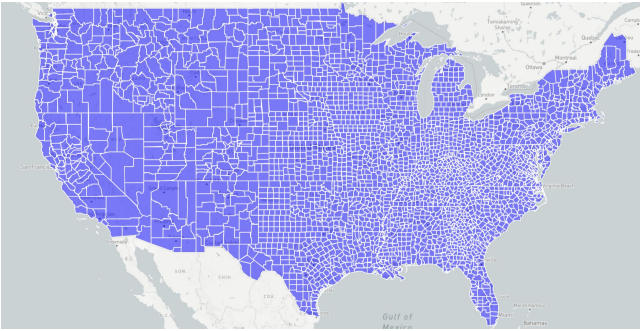


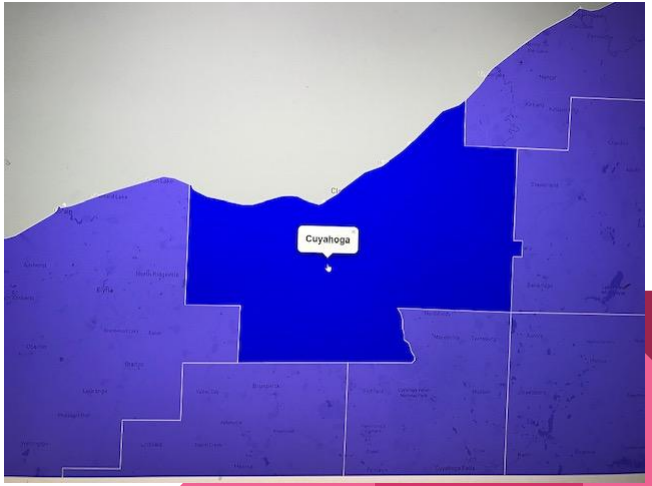
Example of API

Leaflet (Cristina)

1) Leaflet program was able to get counties pulled up on the map, and clicking on counties zoomed in as well as displayed the county name.

2) Next steps with the Leaflet program is to get the election data to pop up on click, and confirm that it is correctly linked with the API data.





Interactive Dashboard (Jermaine)

With integrated data, our end results for an interactive dashboard included:

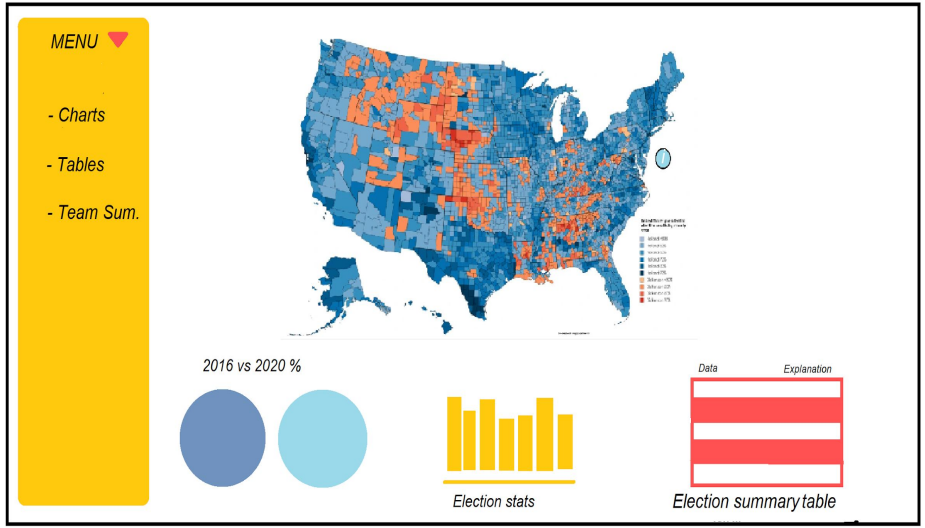
-Interactive sidebar(Javascript)

-Linked pages to maps, team info,etc.

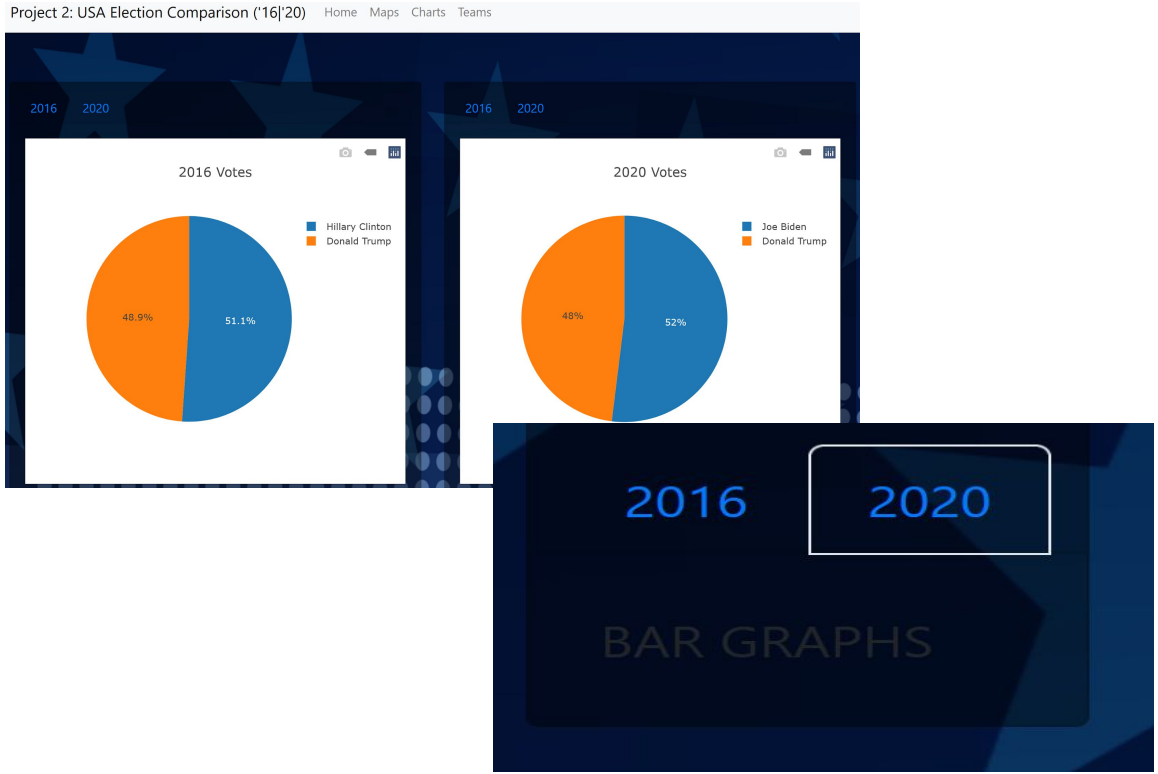
(Flask)

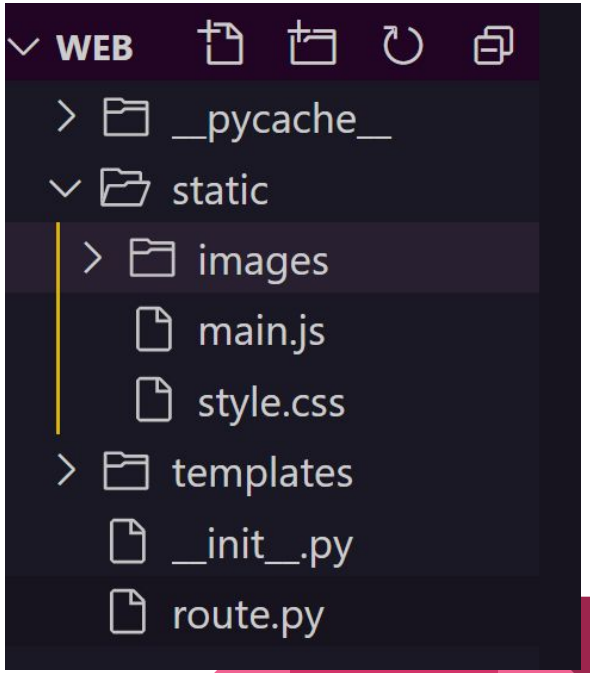
-Used cards to create dashboard layout

(Bootstrap)

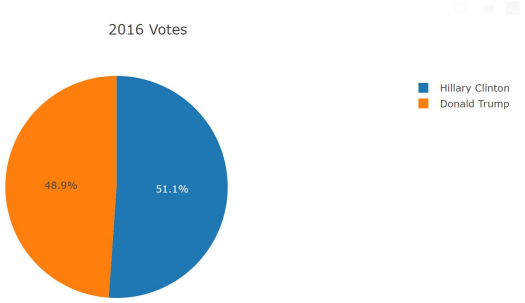


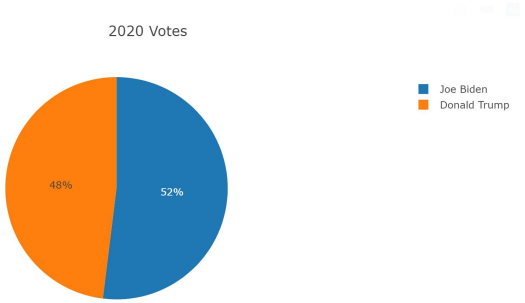
Interactive Dashboard (Jermaine)





Plotly Pie Charts (Leah)





Using Plotly, visualizations were created to show the overall results of the presidential election in both 2016 and 2020. This would give larger context to the votes that were cast at the county level.

Team Page (Leah)

Development Team

Cristina Sheridan

Cristina Sheridan is a research data analyst that loves to play with data ... most days that is:). She has way too many AWS flask moments and looking forward to a future where she will know a bit more:). For this project, she hosted the AWS server and loaded the database, created the map with Leaflet, and contributed to writing the Python apps.

Leah Brady

Leah Brady is a former civil litigator turned data analyst who enjoys telling stories through data. Currently a manager in a legal support company, her belief is that insights from data with change the practice of law. She has skills in Python, SQL, JavaScript, HTML, and CSS. For this project, she created data visualizations using Plotly, wrote the database schema, and contributed to writing the webpage.

Ben Stork

An experienced project manager with a background in data analytics and mechanical engineering. Diverse experience and background in leading and contributing to interdisciplinary teams. Strives to continually acquire knowledge to solve technical issues and break down complex problems into manageable tasks. Always looking to grow and acquire new skills to best contribute and leverage opportunities in a team environment and expand my data analysis knowledge and skills. For this project, he performed the data clean up in a Jupyter Notebook, wrote the Flask app, and contributed to creating Python apps.

Jermaine Coleman

As a recent college graduate, Jermaine is excited to continue strengthening her coding and all around data analyst abilities. Coming in with intermediate HTML/CSS skills, CWRU's boot camp has highly increased Jermaine's web design, data programs, and coding language repertoire. With a knack for visuals and creativity, Jermaine hopes to be promoted to business analyst in her company's finance department. For this project, she created data visualizations, wrote the webpage framework using HTML and Bootstrap, and contributed to writing the Flask app.

Next Steps

1) Complete Leaflet to pull in data from API to show election information when Counties are clicked on.
2) Finalize all website design and confirm that all data is being referenced and shown.
3) Implement AWS or Heroku website so API does not have to be pulled up on local server.