Final Lab - Very Irate Prawns

Voter Analysis

This portion of our final lab will cover the ways in which we will make queries on the voterbase to get analyses and conclusions on their voting patterns. From data like this, we could extrapolate trends based on certain demographics, as well as consider the extent to which the votes cast were influenced by outside factors. Larger-scale pictures of the voting trends such as this will allow us to extract information about which parts are the most vulnerable to influence from outside voter manipulation.

Assuming this Setup

There are a few assumptions that these examples make about model schemas already being written, result sets already queried, as well as a few helper functions being defined:

```
import Mongoose from "mongoose";
import { candidateModel, partyModel, voteModel, voterModel }
  from "./models/models.js";

const candidates = candidateModel.find();
const parties = partyModel.find();
const votes = voteModel.find().sort('-date'); // assuming votes have
const voters = voterModel.find();

const demographics = [[18,29], [30,44], [45,64], [65,undefined]];
const states = ['Alabama','Alaska','American Samoa','Arizona','Arkans

// gets a date a certain number of years before now
//  - if years is undefined, returns the date 150 years before now
const getDateYearsBeforeNow = (years) => {
  let current = new Date();
  current.setFullYear(current.getFullYear() - (years !== undefined ?
  return current;
};
```

```
// Gets the party of the most recent vote for a voter
const getVoterPartyFromMostRecentVote = (voter) =>
  getPartyFromCandidate(getCandidateFromVote(getMostRecentVoteFromVot

const addToArrayIfNotIncluded = (item, array) => !array.includes(item

// Gets the most recent vote for a given voter
const getMostRecentVoteFromVoter = (voter) =>
  voter === undefined ? undefined :
  votes.find(vote => vote.voter_id === voter.id);

// Gets the candidate from a given vote
const getCandidateFromVote = (vote) =>
  vote === undefined ? undefined :
  candidates.find(candidate => candidate.id === vote.candidate_id);

// Gets the party from a given candidate's ticket
const getPartyFromCandidate = (candidate) =>
  candidate === undefined ? undefined :
  parties.find(party => party.name === candidate.party_name);

// Gets the party from a given vote
const getPartyFromVote = (vote) => getPartyFromCandidate(getCandidate

// Gets a tally of democrat, republican, and other party votes for a
const getPartyCount = (voterbase) => {
  let partyCount = { "democrat": 0, "republican": 0, "other": 0 };

  voterbase.forEach(voter => {
    let voteParty = getVoterPartyFromMostRecentVote(voter);

    if (voteParty !== undefined) {
      let partyIndex = voteParty.toLowerCase();
      partyIndex = Object.keys(partyCount).includes(partyIndex) ? par
      partyCount[partyIndex]++;
    }
  });

  return partyCount;
};
```

Query Users by Age

```
// gets an array of party representation by voter age
const getPartyRepresentationByAge = () => {
  const results = [];

  demographics.forEach(([ low, high ]) => {
    voterModel.find({
      age: {
        $gte: getDateYearsBeforeNow(low),
        $lt: getDateYearsBeforeNow(high)
      }})
      .then(queryResult => getPartyCount(queryResult))
      .then(partyCount => results.push({
        ages: [low, high],
        voteCount: partyCount
      }));
  });

  return results;
};
```

The data is returned in the following format:

```
[
  {
    ages: [18, 29],
    voteCount: {
      "democrat": 7,
      "republican": 1,
      "other": 2
    }
  },
  {
    ages: [30, 44],
    voteCount: {
      "democrat": 5,
      "republican": 4,
      "other": 1
```

```
    }
  },
  // Array continues with other demographic ranges...
]
```

- Different Age Brackets
  - Stacked bar graph chart to show Dems/Reps for each
- Per-Candidate Pie Charts with percentage of total vote from each age bracket

Query Users by State and Age

```
// gets an array of party representation on a per state and age basis
const getPartyRepresentationByAgeAndState = () => {
  const results = [];

  demographics.forEach(([ low, high ]) => {
    let resultObject = {
      ages: [low, high],
      states: [],
    };

    states.forEach(state => {
      voterModel.find({
          age: {
            $gte: getDateYearsBeforeNow(low),
            $lt: getDateYearsBeforeNow(high) },
          state: state
        })
        .then(queryResult => getPartyCount(queryResult))
        .then(partyCount => resultObject.states.push({
          state: state,
          voteCount: partyCount
        }));
    });

    results.push(resultObject);
  });

  return results;
```

```
};
```

Data is returned in the following form:

```
[
  {
    ages: [18, 29],
    states: [
      {
        state: "Alabama",
        voteCount: {
          "democrat": 1,
          "republican": 2,
          "other": 0
        }
      },
      {
        state: "Alaska",
        votecount: {
          "democrat": 1,
          "republican": 1,
          "other": 1
        }
      },
      // Array continues with objects for each state...
    ]
  },
  {
    ages: [30, 44],
    states: [
      {
        state: "Alabama",
        voteCount: {
          // Vote Counts...
        }
      },
      // More States...
    ]
  },
  // More demographic ranges...
]
```

- Get US Map with demographics at different age ranges
  - Could probably simulate this result on the slides with some queries + maps from previous elections (five thirty eight?)

Query to Return Votes from Users who have voted for more than one Party

```
// gets an array of all users who have voted for more than one party
const getMultiPartyVoterTrends = () => {
  const results = [];

  voters.forEach(voter => {
    const allVotes = [];
    const partiesVoted = new Set();

    votes.forEach(vote => vote.voter_id === voter.id && allVotes.push
    allVotes.forEach(vote => partiesVoted.add(getPartyFromVote(vote))

    if (partiesVoted.size > 1) {
      const allVotesWithParty = [];
      [...allVotes].reverse().forEach(vote => {
        allVotesWithParty.push({
          ...vote,
          party: getPartyFromVote(vote)
        });
      });

      results.push({
        voter: voter,
        votes: allVotesWithParty
      });
    }
  });

  /*
    Conclusions to make here:


      - Look for the number of times a user's vote changes party fron
```

```
        vote
      - Look for "dem - repub" differential - how far is a given vote
        - Votes for dem increase Y, votes for repub decrease Y
        - Further from 0 means more intensity towards a given positic

  */

  return results;
};
```

This is the form the data will be returned in:

```
[
  {
    voter: {
      id: { $oid : "622fe039fc13ae46440002ab" },
      first_name: "Marlee",
      last_name:"Jozefowicz",
      // ...
    },
    votes: [
      {
        id: 123,
        candidate_id: 321,
        date: "01/02/20",
        //...
        party: "republican"
      },
      {
        id: 456,
        candidate_id: 654,
        date: "01/03/21",
        // ...
        party: "democrat"
      }
    ]
  },
  // ...
]
```

```
// gets all moderate voters - voters who have between 40-60% votes fo
const getModerateVoters = () => {
  const multiPartyVoters = getMultiPartyVotingTrends();
  const moderates = [];

  multiPartyVoters.forEach(voter => {
    const voteParties = { "democrat": 0, "republican": 0 };

    voter.votes.forEach(vote => {
      let party = getPartyFromVote(vote).toLowerCase();
      Object.keys(voteParties).includes(party) && voteParties[party]+
    });

    let { democrat, republican } = voteParties;
    let totalVotes = democrat + republican;
    let democratPercent = democrat / totalVotes;

    democratPercent >= .4 && democratPercent < .6 && moderates.push(v
  });

  return moderates;
}
```