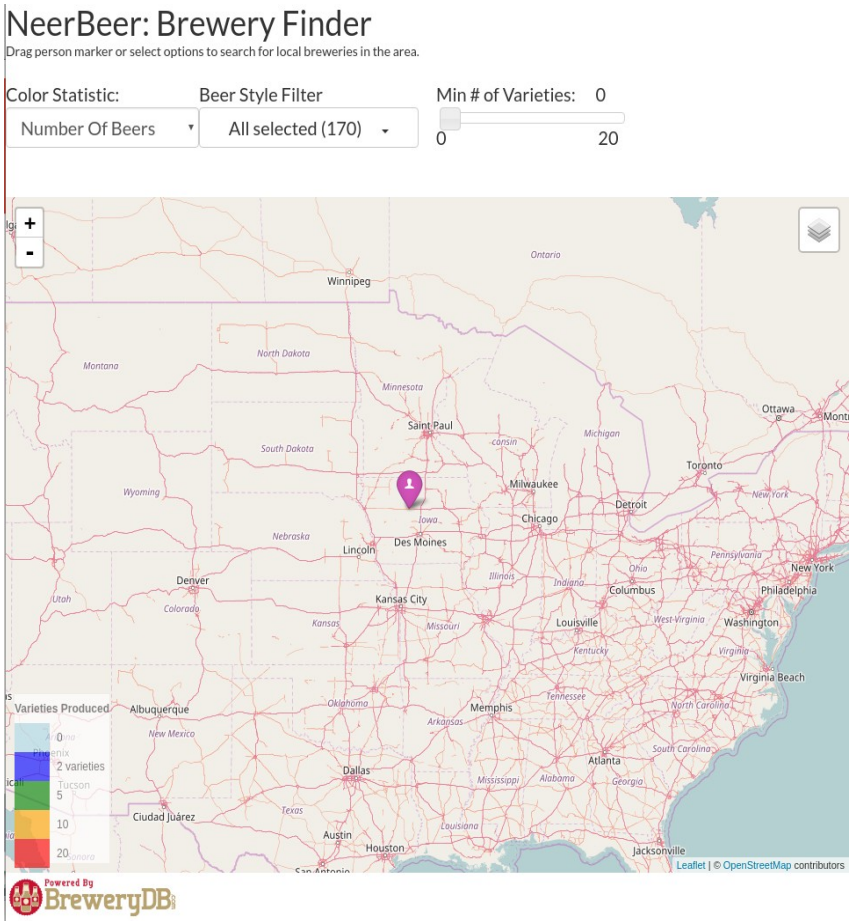
NearBeer

Development Log

-Steven Hadley

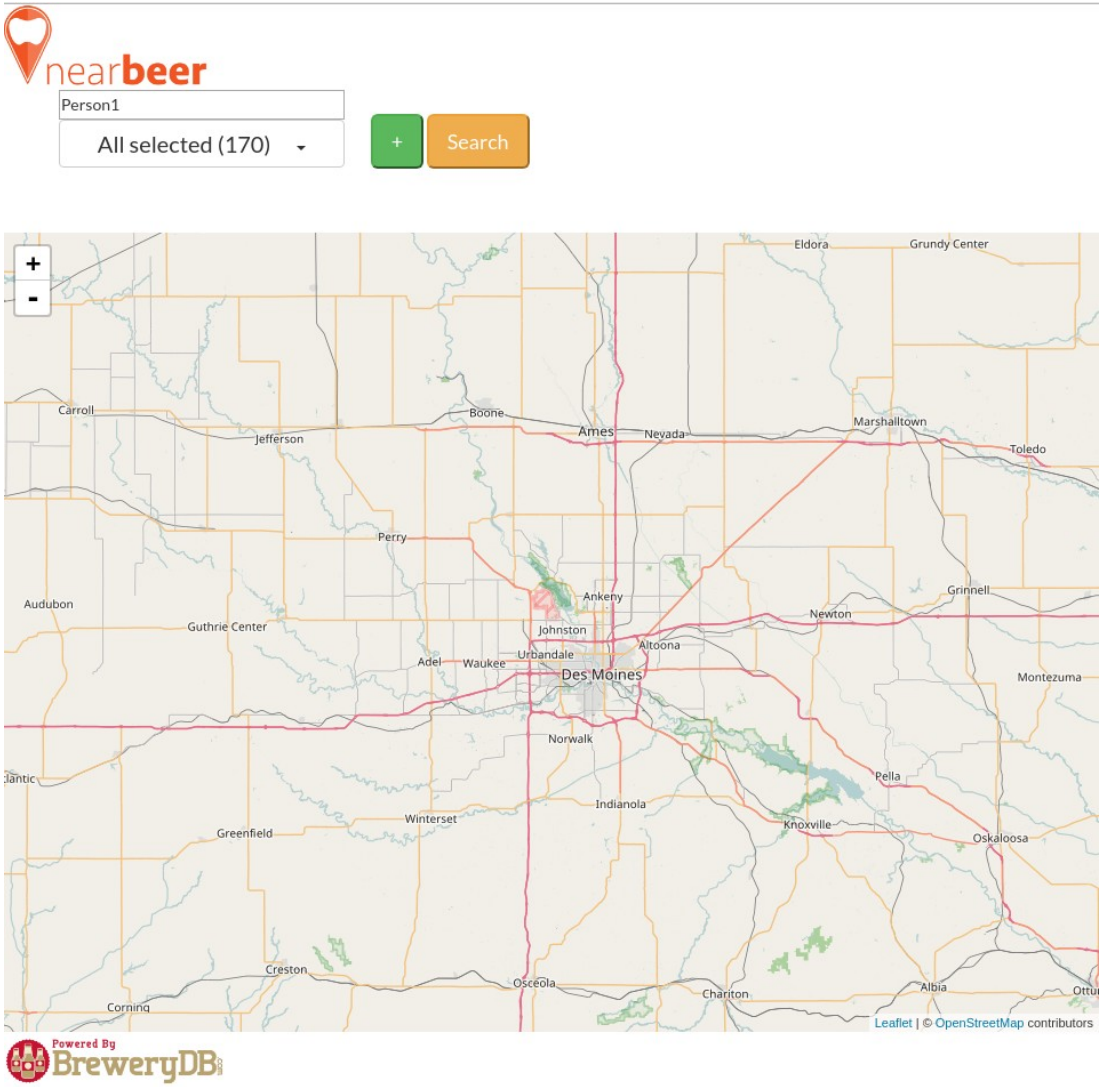
Beginning

- **Goal: Modify Mini Project to make more useful and practical as web app**



Basics

- **Search based on map center and zoom level**
- **Use button to trigger search instead of marker move**
- **Allow for multiple users to select beer styles**



Step 1:
Search Based on Center and Zoom Level

```
function refreshBreweries()
{
    var bounds = Map.getBounds()
    var center = Map.getCenter();
    var rad = Math.round(Math.abs(bounds._southWest.lng - bounds._northEast.lng) * 55);
    if (rad > 100)
    {
        rad = 100;
    }
```

- **Brewery api requires centroid and radius**
- **Use leaflet to populate variables center and bounds**
- **Brewery api has max raidus of 100 miles**

Step 2:
Populate Person Controls

```
$('#addButton').click(() =>
{
    $('#personsDiv').append(
        '<div id="person' + personNumber + '">\
            <input id="name' + personNumber + '" class="personName" value="Person' + personNumber + '"></input><br>\
            <select  id="typeSelect' + personNumber + '" multiple="multiple"> \
            </select> \
        </div>'
    );
    initCombobox('#typeSelect' + personNumber);
    personNumber++;
});
```

- **Increment person number**
- **Use to generate unique DOM id**
- **Add to div inline-block to allow for responsive design**

Step 2:
Populate Person Controls

```
$('#addButton').click(() =>
{
    $('#personsDiv').append(
        '<div id="person' + personNumber + '">\
            <input id="name' + personNumber + '" class="personName" value="Person' + personNumber + '"></input><br>\
            <select  id="typeSelect' + personNumber + '" multiple="multiple"> \
            </select> \
        </div>'
    );
    initCombobox('#typeSelect' + personNumber);
    personNumber++;
});
```

- **Increment person number**
- **Use to generate unique DOM id**
- **Add to div inline-block to allow for responsive design**

Step 3:
Hook up Person Controls

1. Get selected styles from comboboxes

```
function getSelectedStyles()
{
    var optDict = {};
    for(var i = 1; i < personNumber; i++)
    {
        var selector = '#typeSelect'+ i +' option:selected';
        var selectedOpts =  $(selector);
        var optList = [];
        for(var j =0; j < selectedOpts.length; j++)
        {
            optList.push(selectedOpts[j].value);
        }
        var key = $('#name'+i).val();
        optDict[key] = optList;
    }
    return optDict;
}
```

2. Build dictionary of styleids base on user name

```
function getBeerSelection(styles, beers)
{
    var selectDict = {}
    for (var p in styles)
    {
        selectDict[p] = [];
    }

    for (var p in styles)
    {
        for( var b in beers)
        {
            var s =beers[b].styleId;
            if(s != null)
            {
                var personStyle = styles[p];
                var result = $.inArray(s.toString(), personStyle);
                if(result  > -1)
                {
                    selectDict[p].push(beers[b].name);
                }
            }
        }
    }
    return selectDict;
}
```

3. Count matches with dictionary for each brewery

```
function getScore(selections)
{
    var score = 0;
    for (var p in selections)
    {
        if(selections[p].length > 0)
            score++;
    }
    return score;
}
```

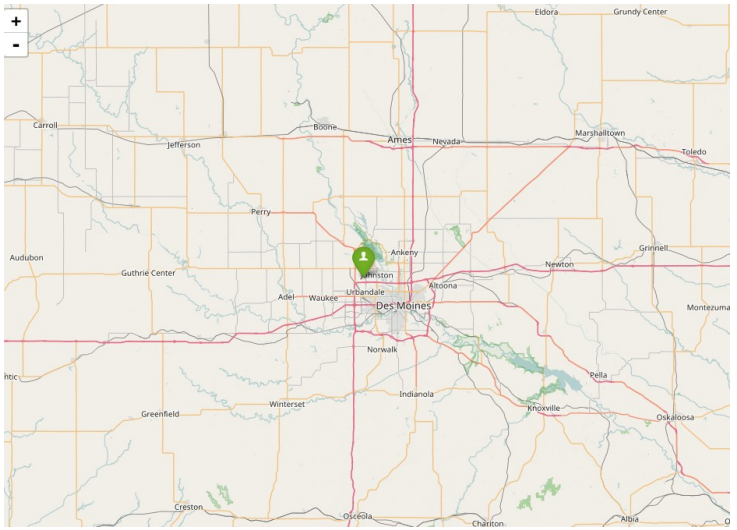
4. Color markers based on users matched by each brewery

```
function getMarkerColors(score)
{
    if(personNumber == 3)
    {
        if(score == 2)
        {
            return darkIcon;
        }
        else
        {
            return lightIcon;
        }
    }
    else if(personNumber == 4)
    {
        if(score == 3)
        {
            return darkIcon;
        }
        else if(score == 2 )
        {
            return mediumIcon;
        }
        else
        {
            return lightIcon;
        }
    }
    else
    {
        if (score == personNumber - 1)
        {
            return darkIcon;
        }
        else if(score == personNumber -2)
        {
```

Step 4: Geolocation

- Followed example from class
- Gotcha was class code only works in https://
- Github pages SSL by default, ElasticBeanstalk not so much
- Went ahead and registered, domain name, and filed with certificate authority
- Cost money ($30), but much easier to share with friends
- Still less than I would have spent in textbooks in most classes

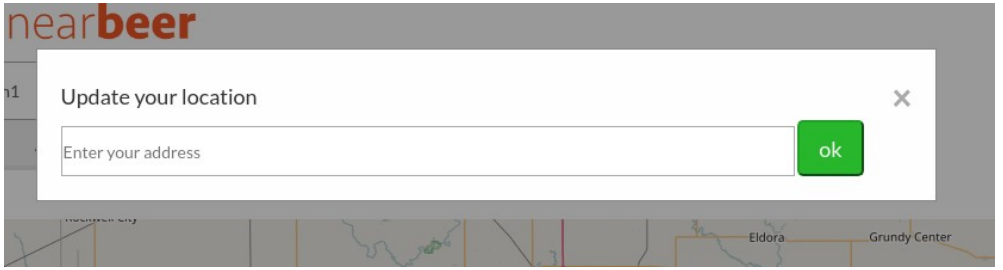
```
function updatePersonMarker(coord)
{
    personMarker.setLatLng(coord);
    Map.panTo(coord);
}

function showPosition(pos)
{
    var coord = new  L.LatLng(pos.coords.latitude, pos.coords.longitude)
    userLocation = coord;
    updatePersonMarker(coord);
};
```



Step 5: Geocoding

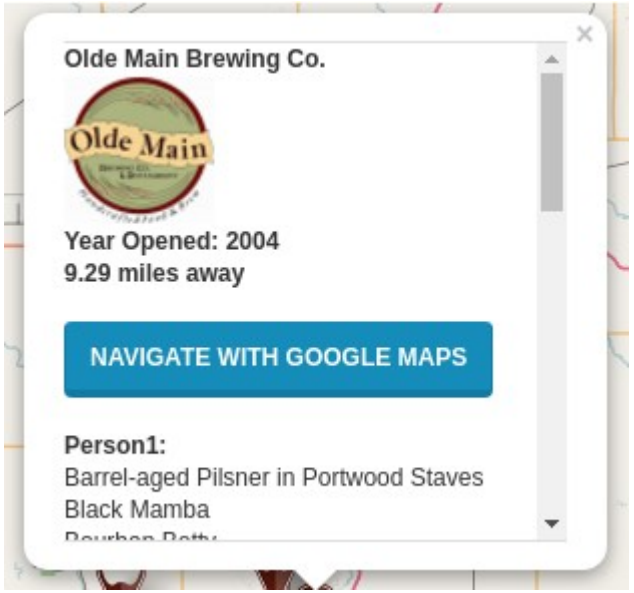
- **Wanted users to be able to update their address**
- **Geolocation doesn't always work**
- **Sometimes you are searching for a trip**
- **Wanted to use user location for calcuating brewery distance and providing directions**

```
function getGeocode(address, callback)
{
    var geocodingAPI_URL =
            'https://maps.googleapis.com/maps/api/geocode/json?address='
            + address
            + '&key=AIzaSyCq9BzYJbmdgg4lweRQiSqdKG2LqX6q_TI'
            + '&sensor=true';
        $.getJSON(geocodingAPI_URL, (json) =>
        {
            var lat = json.results[0].geometry.location.lat;
            var long = json.results[0].geometry.location.lng;
            callback([lat,long]);
        });
}
```



Step 6:
Distance and Directions

- **In proposal, I had planned on using Leaflet Routing machine.**
- **Decided link to google maps was more useful for mobile use cases**
- **Decided to use Turf.js instead to provide distance measurements to help decide which location to visit**



Final Product