

House Hunt

Headline 2

Headline 3

Headline 4

Headline 5

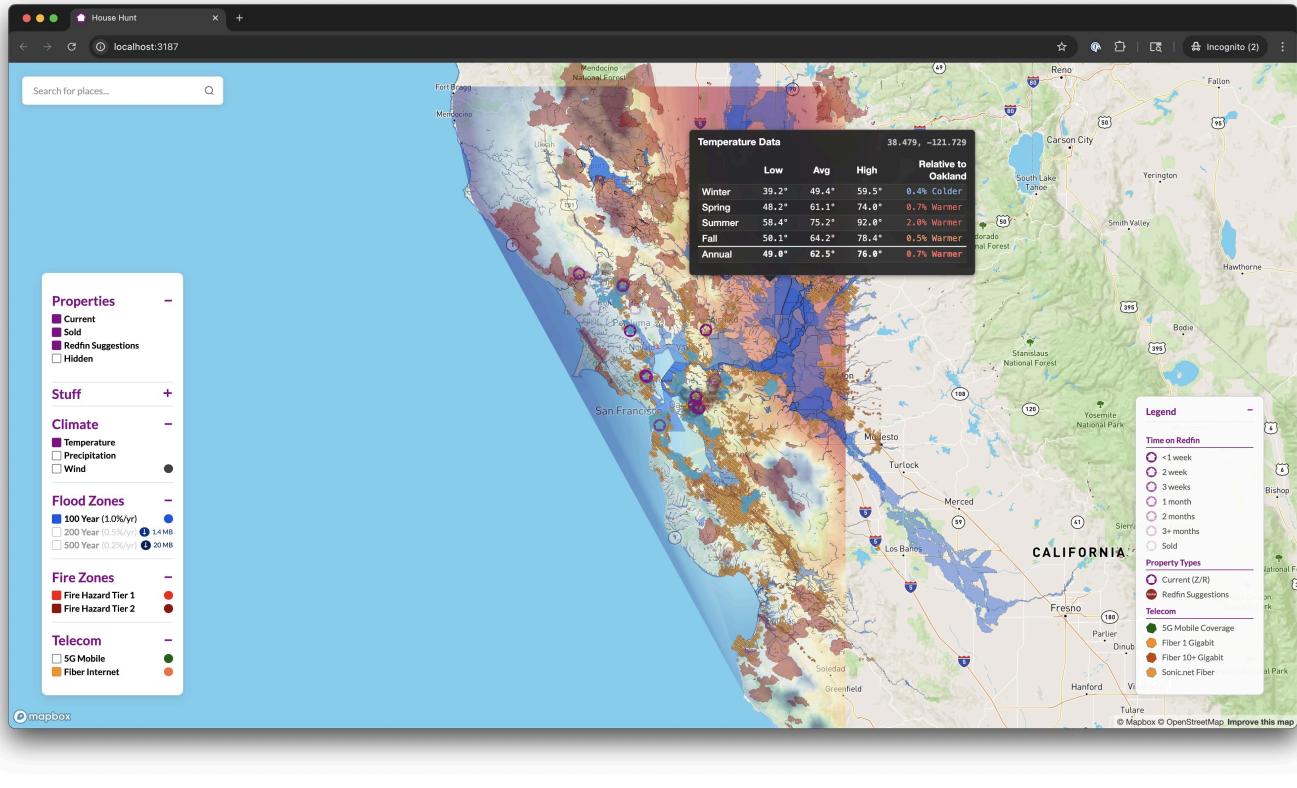
Headline 6

A smart map for finding our new home in the Bay Area.

This is an interactive map to help us house hunt more effectively. It shows the houses we're considering, nearby amenities that matter to us, important risk information like flood zones and fire hazards, and accessibility to 5G wireless and fiber optic internet service.

 [Live Site:](https://max:gracie@househunt.ezzat.com) <https://max:gracie@househunt.ezzat.com>

Note: Credentials are baked into the URL above but, if prompted, use username `max` and password `gracie`.



©2025 Feral Creative / @feralcreative. All rights reserved.

What It Does

Tracks and Maps Potential New Homes

When we find a house on Redfin that looks promising, we just share it to an email alias (houses@ezzat.com) that forwards to a dedicated gmail account (ziadandrebecca@gmail.com).

Every 30 minutes, Google Apps Script <gs/collectHouseListings.gs> checks for new emails from Redfin and adds them to the map.

Email Filtering Criteria:

The system processes two types of Redfin emails:

1. **Shared Listings:** Subject contains `"wants you to see"` (marked Z/R for Ziad/Rebecca)
2. **Redfin Suggestions:** Subject starts with `"A "` or `"An "` and contains `"home for you"` (marked with em dash —)

Processing Rules:

- URLs must be from `redfin.com` or `redf.in` domains
- Redfin URLs must contain `/home` path (ignores generic pages)
- Ignores social media links (Facebook, Twitter, Pinterest)
- Deduplicates against existing properties in the sheet
- **Tracks who suggested each property** based on email type and sender
- Adds them to our map with all the details (price, beds, baths, square footage, etc.)
- Grabs the main photo URL from the listing
- Calculates how long it's been on the market (newer listing markers are more opaque)
- Estimates the total monthly cost including mortgage, taxes, HOA, and insurance

Points of Interest

- **Coffee shops** (Starbucks, Peet's)
- **Retail stores** (Container Store, Total Wine, Walmart)
- **Big box stores** (Target, Costco, Home Depot, Lowe's)
- **Healthcare facilities** (Kaiser medical centers and offices)
- **Grocery stores** (Whole Foods, Safeway, Albertsons, Trader Joe's, Raley's, Bel Air, Nob Hill)
- **Telecom coverage** (5G mobile and fiber internet areas)

Environmental Risks & Climate Data

- **Flood zones** show areas that could flood during major storms
- **Fire hazard zones** highlight areas with higher wildfire risk
- **Climate data** displays temperature, precipitation, and wind patterns with interactive hover popups
- **Relative comparisons** show how each location compares to our North Oakland home using quintile-based rankings
- All data comes from official government sources (FEMA, CAL FIRE, NOAA, and NREL)

Real Monthly Cost Estimates

For each property, we can see the true monthly cost including:

- Mortgage payments (using 15-year fixed mortgage rates from [Bankrate.com](#))
- Property taxes (with Prop 19 considerations)
- HOA fees, if applicable
- Insurance costs (including [FAIR Plan](#) insurance for high-risk areas)

How to Use It

Finding Properties

1. **Browse Redfin** like you normally would
2. **Share interesting listings** using the native Redfin share button to houses@ezzat.com
3. **Check the map** - new properties appear automatically within a few minutes
4. **Click any house marker** to see details, photos, and calculated monthly costs

Understanding the Map

- **Brighter markers** = newer listings (just hit the market)
- **Faded markers** = been on the market for weeks/months
- **Gray markers** = sold properties (for reference)
- **Property popups** show who suggested each property (Z=Ziad, R=Rebecca, --=Redfin direct suggestions)
- **Toggle layers** using the controls to show/hide different types of places
- **Legend sections** are collapsible - click any section header to expand/collapse
- **Default view** shows Properties and Time on Redfin sections open, others collapsed
- **Your preferences** are saved automatically and persist between visits
- **Clean up button** quickly resets the map to show only current properties, turning off all other layers

Backend Operations

How It All Works

I set up an email alias (`houses@ezzat.com`) for sharing Redfin listings, and a Google Apps Script to watch for incoming Redfin listings. When either of us shares a property to that address, the system:

1. Recognizes it's a Redfin listing (ignores spam and other emails)
2. Extracts the property URL from the email and adds it to our Google Sheet
3. `redfinUpdateCurrent.gs` and `redfinUpdateAll.gs` visit the Redfin pages to collect detailed property data
4. Calculates the monthly costs using current rates and tax info
5. Shows it on the map within a few minutes

The Email-to-Data Process

Step 1: Email Detection Every 30 minutes, a Google Apps Script checks our shared Gmail inbox for new emails from `donotreply@redfin.com` that contain “wants you to see” in the subject line. This filters out all the other Redfin emails (price changes, saved searches, etc.) and only processes the ones where we actually shared a specific property.

Step 2: URL Extraction and Storage `collectHouseListings.gs` reads the email body and looks for Redfin property URLs. It's smart enough to ignore social media links, generic Redfin pages, and other junk - it only grabs the actual property listing URLs. It also figures out who sent the email (`Ziad wants you to see` vs. `Rebecca wants you to see`) based on the subject line format, then adds the URL to our Google Sheet with basic metadata including the **Suggested By** field that tracks who found each property.

Step 3: Data Scraping with Multiple Scripts Two different scripts handle data extraction based on urgency and scope:

`redfinUpdateCurrent.gs` (runs every 12 hours) focuses on active listings and extracts over 20 different pieces of information:

- **Basic details:** Price, beds, baths, square footage, lot size, year built
- **Market info:** Days on market, price history, HOA dues
- **Property features:** Garage spaces, basement, pool, etc.
- **Photos:** Grabs the main listing photo
- **Location data:** Converts the address to map coordinates

`redfinUpdateAll.gs` (batch processing) handles comprehensive updates for all properties, including sold ones, processing 100 records at a time to avoid timeouts and ensure complete data coverage.

Step 4: Smart Data Processing The system is designed to handle Redfin's quirks - like how they sometimes show “Coming Soon” instead of a price, or how square footage might be listed in different places on the page. It tries multiple ways to find each piece of data and falls back gracefully when something isn't available.

Property Suggestion Tracking

The system automatically tracks property sources using two different systems:

1. Shared Listings (Manual Shares)

- **Email Pattern:** Subject contains "wants you to see"
- **Name Detection:** Looks for "Ziad" or "Rebecca" in subject line
- **Markers:** Z (Ziad), R (Rebecca), or blank if no name detected
- **Examples:**
 - "Ziad wants you to see this home" → Found: Z
 - "Rebecca wants you to see this home" → Found: R

2. Redfin Direct Suggestions (Algorithm-Generated)

- **Email Pattern:** Subject starts with "A " or "An " and contains "home for you"
- **Marker:** – (em dash) for all Redfin algorithm suggestions
- **Examples:**
 - "A home for you in Oakland" → Found: –
 - "An updated home for you in Berkeley" → Found: –

Map Display:

- **Regular properties:** Use standard property icon (Z/R found values)
- **Redfin suggestions:** Use special Redfin icon (– found value)
- **Separate toggles:** Independent checkboxes to show/hide each type
- **Default state:** Both property types visible by default for new visitors

This dual-tracking system helps distinguish between properties we manually discovered vs. ones Redfin's algorithm suggested to us!

The Data Sources

- **Property info:** Automatically scraped from Redfin listings
- **Monthly costs:** Calculated using our [mortgage calculator spreadsheet](#)
- **Climate data:** [NOAA Climate Normals](#) (2006-2020 period)
- **Wind data:** [NREL Wind Integration National Dataset](#) (2001-2020 period)
- **Flood zones:** Official FEMA flood maps
- **Fire hazards:** [California fire risk data](#)
- **Broadband coverage:** [FCC nationwide broadband data](#)
- **Points of interest:** Manually curated locations we care about



Map Markers Reference

Property Markers

Active Properties (Age-Based Opacity)

Property age is calculated using "Days on Redfin" data automatically extracted from each listing page, providing accurate market timing information.

Icon	Age Range	Description
	< 1 week	Newest listings (100% opacity)
	1-2 weeks	Recent listings (85% opacity)
	2-3 weeks	Moderately recent (70% opacity)
	1 month (4-7 weeks)	Month-old listings (55% opacity)
	2 months (8-11 weeks)	Two-month-old listings (40% opacity)
	3+ months (12+ weeks)	Oldest active listings (25% opacity)
	Sold Properties	Properties marked as sold (grayscale filter + 15% opacity)
	Redfin Suggestions	Properties suggested by Redfin's algorithm (— found value)
	Hidden Properties	Properties with "Hide" column checked in the Google Sheet
	Home	Our current home in North Oakland

Healthcare

Icon	Type	Description
	Kaiser Medical Centers	Large comprehensive facilities with hospitals
	Kaiser Medical Offices	Smaller outpatient clinics and primary care

Coffee Shops

Icon	Type	Description
	Starbucks	Starbucks Coffee locations
	Peet's Coffee	Peet's Coffee & Tea locations

Retail Stores

Icon	Type	Description
	Target	Target retail stores ¹
	Container Store	The Container Store locations
	Costco	Costco Wholesale warehouses
	Total Wine	Total Wine & More stores
	Walmart	Walmart retail locations ²

Hardware Stores

Icon	Type	Description
	Ace Hardware	Ace Hardware stores
	Home Depot	The Home Depot locations
	Lowe's	Lowe's Home Improvement stores

Grocery Stores

Icon	Type	Description
	Albertsons	Albertsons grocery stores
	Whole Foods	Whole Foods Market locations
	Safeway	Safeway grocery stores
	Trader Joe's	Trader Joe's grocery stores
	Raley's	Raley's supermarkets
	Bel Air	Bel Air Markets
	Nob Hill	Nob Hill Foods stores

Climate Data

Climate data helps understand long-term weather patterns for each area. The map displays temperature and precipitation using smooth heatmap overlays with comprehensive hover information.

Layer	Data Type	Description
	Temperature	Shows average temperature patterns across the region using NOAA climate normals (2006-2020)
	Precipitation	Shows annual precipitation patterns using NOAA climate normals (2006-2020)
	Wind	Shows wind speed and direction patterns using NREL wind data (2001-2020)

Hover Information

Temperature

- Annual, Summer, Winter temperature data
- Low (minimum), Avg (average), High (maximum) values for each period
- All temperatures shown in degrees Fahrenheit

Precipitation

- Annual total precipitation in inches
- Summer and Winter seasonal averages

Wind

- Seasonal wind speeds (Winter, Spring, Summer, Fall) in miles per hour (converted from m/s)
- Prevailing wind directions using 16-point compass rose (N, NNE, NE, etc.)
- Vector-averaged directions (meteorologically correct averaging)
- Visual wind arrows showing both speed (color intensity) and direction on the map
- Relative windiness compared to North Oakland home location using quintile-based rankings

Climate data comes from [NOAA Climate Normals](#) for temperature/precipitation (2006-2020) and [NREL Wind Integration National Dataset](#) for wind patterns (2001-2020).

Climate Comparison System

The climate popups include a “Relatively” column that compares each location’s climate to our current home in North Oakland (37.854, -122.271). This provides personally meaningful context for house hunting decisions.

How It Works

Reference Point Selection:

- System finds the closest NOAA climate data point to our North Oakland home coordinates
- All comparisons use this single reference point for consistency
- Provides “compared to home” context rather than abstract regional rankings

Quintile-Based Comparison Logic:

The system calculates quintiles (5 equal groups) from all climate data points in the region, then compares each location's quintile position to our home location's quintile position.

```
// Temperature & Precipitation Quintile Logic
Much Colder/Dryer: ≥ 2 quintiles below home
Colder/Dryer:      1 quintile below home
Similar:           Same quintile as home
Warmer/Wetter:     1 quintile above home
Much Warmer/Wetter: ≥ 2 quintiles above home
```

Why Quintiles?

- Data-driven:** Based on actual distribution of all regional climate data
- Relative context:** Shows where each location ranks compared to the full range of options
- Consistent scale:** Same ranking system applies to both temperature and precipitation
- Meaningful differences:** Each quintile represents a significant 20% segment of the data range

Example Climate Popups

Temperature Popup:

Temperature Data 37.979, -122.479				
	Low	Avg	High	Relatively
Winter	44°	51°	58°	Similar
Spring	48°	56°	65°	Similar
Summer	54°	64°	73°	Colder
Fall	52°	61°	69°	Similar
Annual	50°	58°	66°	Similar

Precipitation Popup:

Precipitation Data 37.979, -122.479		
	Avg	Relatively
Winter	5.2"	Much Wetter
Spring	2.1"	Dryer
Summer	0.1"	Similar
Fall	3.8"	Wetter
Annual	11.2"	Much Wetter

Wind Popup:

Wind Data 37.979, -122.479			
	Speed	Direction	Relatively
Winter	7.2	NW	Windier
Spring	9.2	NW	Much Windier
Summer	10.7	WNW	Windier
Fall	6.5	NW	Similar
Annual	8.4	NW	Windier

Speed in mph • Direction shows prevailing wind
Vector-averaged using meteorologically correct methods

This system answers the practical question: "How will this location's weather compare to what we're used to at home?"

Telecom Coverage

Good internet is essential for working from home. The map shows coverage areas using hexagonal overlays:

5G Mobile Coverage

Icon	Coverage Type	Description
	5G Mobile	5G NR mobile broadband coverage areas

Fiber Internet Coverage

Icon	Speed Tier	Description
	1 Gigabit	Standard gigabit fiber connections
	10+ Gigabit	Multi-gigabit fiber connections
	Sonic.net	Sonic.net fiber service (highlighted with blue border)

Data comes from the **FCC's nationwide broadband database**. Hexagons represent H3 geographic coverage areas.

Marker Behavior

- **Dynamic Sizing:** All markers scale with map zoom level (larger when zoomed in)
- **Z-Index Layering:** Properties on top, then Peet's over Starbucks, then other POIs
- **Age-Based Opacity:** Property markers fade over time (newest listings are most opaque)
- **Interactive Popups:** Click any marker for detailed information
- **Group Toggle:** Control visibility by category (Coffee, Retail, etc.) or individual type
- **Smart Property Filtering:** Property visibility follows logical rules:
 - **Current only:** Shows current properties (Z/R found)
 - **Sold only:** Shows sold properties (Z/R found)
 - **Redfin only:** Shows current Redfin suggestions
 - **Current + Sold:** Shows all regular properties (current and sold)
 - **Sold + Redfin:** Shows all Redfin properties (current and sold suggestions)
 - **All checked:** Shows all property types
- **Authentic Icon Shapes:** Markers respect their original SVG shapes (Redfin icons show less rounded corners)

Resources

Design & Data

- **Figma Design File:** [ezzat.com Design System](#) - Contains all custom icons, graphics, and design assets
- **Property Database:** [House Hunt Listings](#) - Live Redfin property data and configuration
- **Mortgage Calculator:** [Monthly Cost Calculator](#) - Detailed monthly payment calculations including Prop 19³ considerations

External APIs & Services

- **Mapbox GL JS:** Interactive map rendering and controls
- **Google Sheets API:** Real-time data synchronization
- **Google Apps Script:** Automated data collection and processing
- **FEMA National Flood Hazard Layer:** Environmental risk data
- **CAL FIRE Hazard Severity Zones:** Fire risk assessment data
- **NOAA Climate Normals:** Temperature and precipitation data
- **NREL Wind Integration National Dataset:** High-resolution wind speed and direction data
- **FCC Nationwide Broadband Data:** Telecom coverage information



Customization Guide

Adding New POI Types

1. Add data to `/data/stuff.json`
2. Create icon in `/img/icons/`
3. Update POI groups in `js/houseHunt.js`
4. Add checkbox controls in `index.html`

Customizing Property Data

1. Update Google Sheets column headers
2. Modify server endpoint in `server/server.js`
3. Update popup template in `js/houseHunt.js`

Adding Environmental Layers

1. Process data to GeoJSON format
2. Add files to `/data/` directory
3. Implement loading logic in `js/houseHunt.js`
4. Add UI controls in `index.html`

Clean Up Button

The “Clean up” button provides a quick way to reset the map to a clean state showing only current properties. Located at the bottom of the control panel, it:

- **Turns on:** Current properties checkbox
- **Turns off:** All other property types (Sold, Redfin Suggestions, Hidden)
- **Turns off:** All POI categories (Coffee, Retail, Hardware, etc.)
- **Turns off:** All environmental layers (Climate, Fire, Telecom)
- **Persists changes:** Saves all checkbox states to localStorage
- **Fast tooltip:** Custom tooltip appears in 100ms instead of browser default ~500ms

Architecture

- **Frontend:** Vanilla JavaScript with Mapbox GL JS for the interactive map, SCSS for styling
- **Backend:** Node.js/Express server that serves the frontend and provides API endpoints
- **Database:** Google Sheets for property data
- **Automation:** Google Apps Script for email monitoring and data collection
- **Deployment:** PM2 on a Synology NAS

UI/UX Improvements

- **Map Scale Positioning:** Scale control repositioned next to Mapbox logo for cleaner layout

- **Marker Shape Fidelity:** Removed CSS border-radius override to preserve authentic SVG icon shapes
- **Property Visibility Logic:** Implemented smart filtering where property types only show when their respective checkboxes are enabled
- **Fast Tooltips:** Custom tooltip system with 100ms delay for better responsiveness

How the Email Automation Works

Collection Script (Google Apps Script):

1. **Email Monitoring:** Checks `ziadandrebecca@gmail.com` every 30 minutes
2. **Filtering:** Only processes emails from `donotreply@redfin.com` with "wants you to see" in the subject
3. **URL Extraction:** Pulls the property URL from the email body, avoiding social media links
4. **Deduplication:** Maintains a list of processed URLs to avoid duplicates
5. **Basic Storage:** Adds the URL and sender info to the Google Sheet

Data Scraping Scripts:

6. `redfinUpdateCurrent.gs` : Fast updates for active listings only - skips sold properties, runs every 12 hours
7. `redfinUpdateAll.gs` : Comprehensive updates for all properties - processes 100 records per run, runs every 10 minutes, cycles through entire database
8. `redfinExtractors.gs` : Contains shared utility functions for geocoding and data extraction
9. **Sheet Updates:** All scripts populate the detailed property data in the Google Sheet

Script Timing & Strategy

`redfinUpdateCurrent.gs` - Frequent Updates (Every 12 Hours)

- **Scope:** Only processes properties where `Sold = FALSE`
- **Speed:** Fast execution since it skips sold properties
- **Purpose:** Keep active listings fresh with current prices, status changes, and new photos
- **Why frequent:** Active listings change rapidly (price drops, status updates, new photos)

`redfinUpdateAll.gs` - Comprehensive Cycles (Batch Processing)

- **Scope:** Processes ALL properties in the database (sold and active)
- **Speed:** Slower, handles 100 properties per run to avoid timeouts
- **Purpose:** Backfill missing data, update historical records, catch edge cases
- **Why less frequent:** Sold properties rarely change, comprehensive updates are resource-intensive

Key Files

- `gs/collectHouseListings.gs` - Email monitoring and URL extraction
- `gs/redfinUpdateAll.gs` - Main data scraping orchestrator
- `gs/redfinExtractors.gs` - Individual data extraction functions
- `server/server.js` - Express server with Google Sheets API integration
- `js/houseHunt.js` - Frontend map logic, marker management, and climate popup functionality
- `css/househunt.scss` - All the styling including marker opacity rules and climate popup styling

Developer Utilities

The `utils/` directory contains specialized tools for data processing, conversion, and analysis:

Environmental Data Processing

- `splitFireData.js` - Splits large CAL FIRE GeoJSON files by geographic region for better performance
- `splitFireDataStream.js` - Memory-efficient version for processing massive fire datasets
- `trimFireDataGeographic.js` - Trims fire hazard data to Bay Area boundaries
- `splitFloodplains.js` - Processes FEMA flood zone data into manageable chunks
- `splitFloodplainsStream.js` - Streaming version for large flood datasets
- `trimEastOfManteca.js` - Geographic trimming utility for eastern boundary cleanup

Climate Data Processing

- `extract-climate-geojson.py` - Extracts and processes NOAA climate normals from NetCDF files to GeoJSON
- `generate-climate-rasters.py` - Generates smooth PNG heatmap overlays from climate GeoJSONs
- `climate-layer-config.py` - Creates configuration files and templates for climate layer integration

Wind Data Processing

- `wind_extract_base.py` - Core wind data processing functions and utilities
- `wind_extract_weekly.py` - Processes hourly NREL wind data into weekly climatology (2001-2020)
- `wind_extract_monthly.py` - Creates monthly wind climatology aggregates
- `wind_extract_daily.py` - Generates daily wind pattern summaries
- `wind_extract_hourly.py` - Processes raw hourly wind data with quality control
- `wind_extract_ampm.py` - Processes wind data into AM/PM temporal aggregations
- `wind_extract_geojson.py` - Converts processed wind data to GeoJSON format
- `wind_extract_geojson_enhanced.py` - Enhanced GeoJSON generation with improved metadata
- `wind_extract_geojson_netcdf.py` - Direct NetCDF to GeoJSON conversion for wind data
- `wind_generate_png_raster.py` - Creates georeferenced PNG raster overlays from wind GeoJSON data
- `wind_prepare_hourly_data.py` - Prepares and validates hourly wind data for processing
- `wind_process_enhanced_pipeline.py` - Comprehensive wind data processing pipeline
- `boundary_filter.py` - Removes ocean-based wind stations using hybrid land/water detection

Telecom Data Processing

- `convertFiberCsvToGeojson.js` - Converts FCC fiber-to-premises CSV data to GeoJSON
- `trimTelecom5GCloverdale.js` - Trims 5G coverage data to specific geographic regions

POI Data Collection

- `findStuff.js` - Google Maps Places API integration for fetching POIs
- `safeway.py` - Python script for collecting Safeway locations across California cities

Data Analysis & Extraction

- `analyzeFireData.js` - Analyzes fire risk datasets for coverage statistics and data quality
- `extractCa.py` - Extracts California-specific data from national datasets
- `extractSample.py` - Creates sample datasets for testing and development
- `convertGpkgWithOgr.js` - Converts GeoPackage files to GeoJSON using OGR/GDAL

Usage Examples

```
# Process fire data for Bay Area
node utils/splitFireData.js california-high-hazard-zones.geojson

# Generate climate heatmap overlays
python3 utils/generate-climate-rasters.py --period 2006-2020

# Extract climate data from NetCDF files
python3 utils/extract-climate-geojson.py --variable tave --period 2006-2020

# Process wind data into weekly climatology
python3 utils/wind_extract_weekly.py --input-file data/climate/wind/norcal-wind-2001-2020-hourly.cdf

# Generate wind raster overlays with proper georeferencing
python3 utils/wind_generate_png_raster.py --input-file data/climate/norcal-climate-wind-weekly-2001-2020.cdf

# Process enhanced wind data pipeline
python3 utils/wind_process_enhanced_pipeline.py --input-file data/climate/wind/norcal-wind-2001-2020.cdf

# Extract wind data directly from NetCDF files
python3 utils/wind_extract_geojson_netcdf.py --input-file data/climate/wind/norcal-wind-2001-2020.cdf

# Convert telecom CSV to GeoJSON
node utils/convertFiberCsvToGeojson.js

# Collect POI data via Google Maps API
node utils/findStuff.js

# Analyze dataset coverage
node utils/analyzeFireData.js
```

Environmental Data Processing Pipeline

The flood and fire risk layers required significant data processing:

- **FEMA flood data:** Downloaded massive GeoJSON files, split by zone type, trimmed to Bay Area
- **CAL FIRE data:** Similar process for fire hazard severity zones
- **Processing tools:** Located in `utils/` directory with both standard and streaming versions

Wind Data Processing Pipeline

The wind layer required extensive processing of NREL's high-resolution meteorological data:

Data Acquisition

- **Source:** NREL Wind Integration National Dataset (WIND Toolkit)
- **Coverage:** 2001-2020 (20 years) at hourly resolution
- **Spatial Resolution:** ~2km grid spacing across Northern California
- **Raw Data Size:** ~42GB for the consolidated hourly CSV file
- **Download Process:** Automated via `download_wind_zips.py` using NREL's API

Processing Stages

1. **Data Consolidation:** Individual station/year CSV files merged into master hourly dataset
2. **Geographic Filtering:** Stations filtered to Northern California bounds (Ukiah to Salinas)
3. **Land/Ocean Filtering:** Ocean-based stations removed using hybrid boundary detection
4. **Temporal Aggregation:** Hourly data processed into weekly climatology using vector averaging
5. **Statistical Analysis:** Seasonal, monthly, and annual wind patterns calculated
6. **GeoJSON Generation:** Final data exported as compressed GeoJSON with full metadata

Vector Wind Averaging

Wind data requires special mathematical treatment since direction is circular:

- **U/V Components:** Wind speed/direction converted to eastward (U) and northward (V) components
- **Component Averaging:** U and V components averaged separately over time periods
- **Reconversion:** Averaged components converted back to speed and direction
- **Meteorological Accuracy:** Ensures prevailing wind directions are mathematically correct

Output Formats

- **Full Dataset:** [norcal-climate-wind-weekly-2001-2020.geojson.gz](#) (~37.9MB, 2,182 stations with seasonal/annual averages)
- **Monthly Dataset:** [norcal-climate-wind-monthly-2001-2020.geojson.gz](#) (~4.2MB, monthly aggregations)
- **PNG Rasters:** Georeferenced PNG overlays with world files (.pgw) for precise map alignment
- **Metadata Files:** JSON metadata with processing details, data bounds, and statistics

Wind Visualization Features

- **Black Color Scheme:** Wind data uses shades of black for better contrast against the base map
- **Land-Only Coverage:** Ocean-based wind stations filtered out using hybrid boundary detection
- **High Resolution:** 1500×1500 pixel PNG rasters for detailed visualization
- **Proper Georeferencing:** World files (.pgw) ensure accurate alignment with map coordinates
- **Interactive Popups:** Hover over wind arrows to see detailed speed/direction data with relative comparisons to home location

Getting Started (Technical)

1. **Prerequisites:** Node.js, Google Cloud Project with Sheets API, Mapbox account
2. **Clone and install:** `git clone` → `npm install` in root and server directories
3. **Configuration:** Add service account JSON, update Mapbox token, set Google Sheets ID
4. **Run:** `cd server && node server.js` for development, or `pm2 start ecosystem.config.js` for production

Adding New Features

- **New POI types:** Add data to `data/stuff.json`, create icon, update `js/houseHunt.js`
- **New property fields:** Update scraping functions in `gs/redfinExtractors.gs`, modify popup template
- **New environmental layers:** Process data to GeoJSON, add loading logic to frontend

CSS Z-Index Reference

The following table shows all z-index values used in the application, ordered from highest to lowest:

Z-Index	Element	Purpose	Location
15000	.mapboxgl-ctrl-top-right, .mapboxgl-ctrl-bottom-left	Mapbox navigation and scale controls	css/househunt.scss:18
12001	#search-results	Search dropdown results	css/househunt.scss:13
12000	#search-container	Search input box	css/househunt.scss:66
12000	#layer-controls	Layer toggle controls (left side)	css/househunt.scss:67
12000	#property-legend	Property legend (bottom right)	css/househunt.scss:13
11000	.mapboxgl-popup	Property and marker popups	css/househunt.scss:18
10500	.mapboxgl-popup.climate-hover-popup	Climate data hover tooltips	css/househunt.scss:19
10000	.mapboxgl-popup.layer-popup	Layer information popups	css/househunt.scss:19
1000	.cleanup-button[title]:::before	Clean up button fast tooltip	css/househunt.scss:15
999	.mapboxgl-ctrl	Default Mapbox controls (before fix)	Mapbox GL JS default
100	.mapboxgl-popup	Default Mapbox popups (before fix)	Mapbox GL JS default
10	.mapboxgl-marker	Default Mapbox markers	Mapbox GL JS default
0	.mapboxgl-canvas-container	Map canvas (base map, layers)	Mapbox GL JS default

Notes:

- All z-index values use `!important` to ensure proper stacking
- Mapbox controls were given the highest z-index (15000) to ensure visibility
- Interactive elements (search, controls) are above informational elements (popups)
- Climate hover tooltips are between regular popups and layer popups for proper interaction

Footnotes

1. **Target:** Yes, we are currently boycotting Target because they are [hot garbage.](#) ↩
2. **Walmart:** Walmart is also hot garbage, of course, but are they really worse than Amazon at this point? ↩
3. **Proposition 19:** California's Proposition 19 (passed in 2020) allows homeowners over 55, disabled persons, or disaster victims to transfer their property tax basis to a new home anywhere in the state. This is relevant to us because we can potentially take our current low property tax assessment with us when we move, significantly reducing the property taxes on a new home. ↩