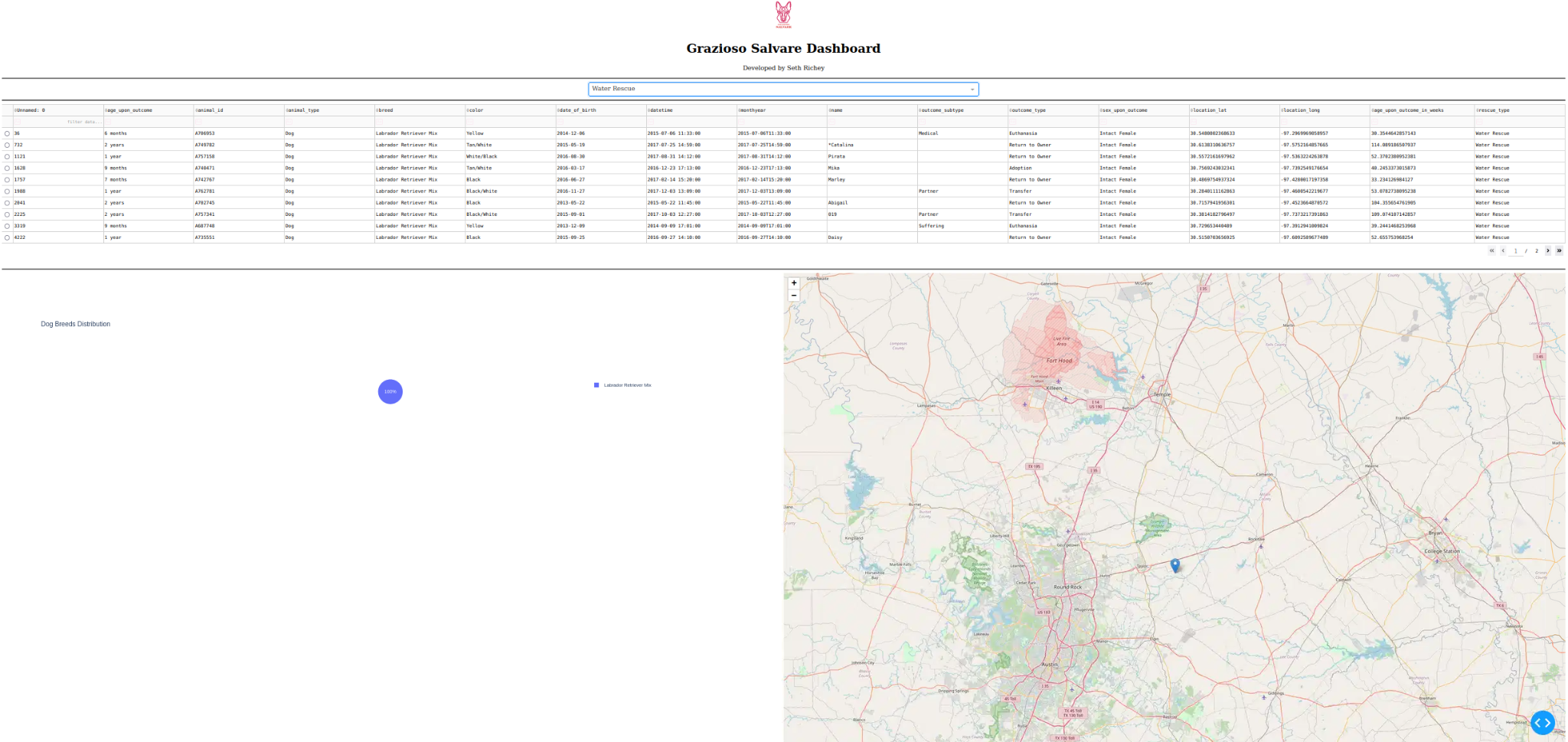
## **Project Functionality**

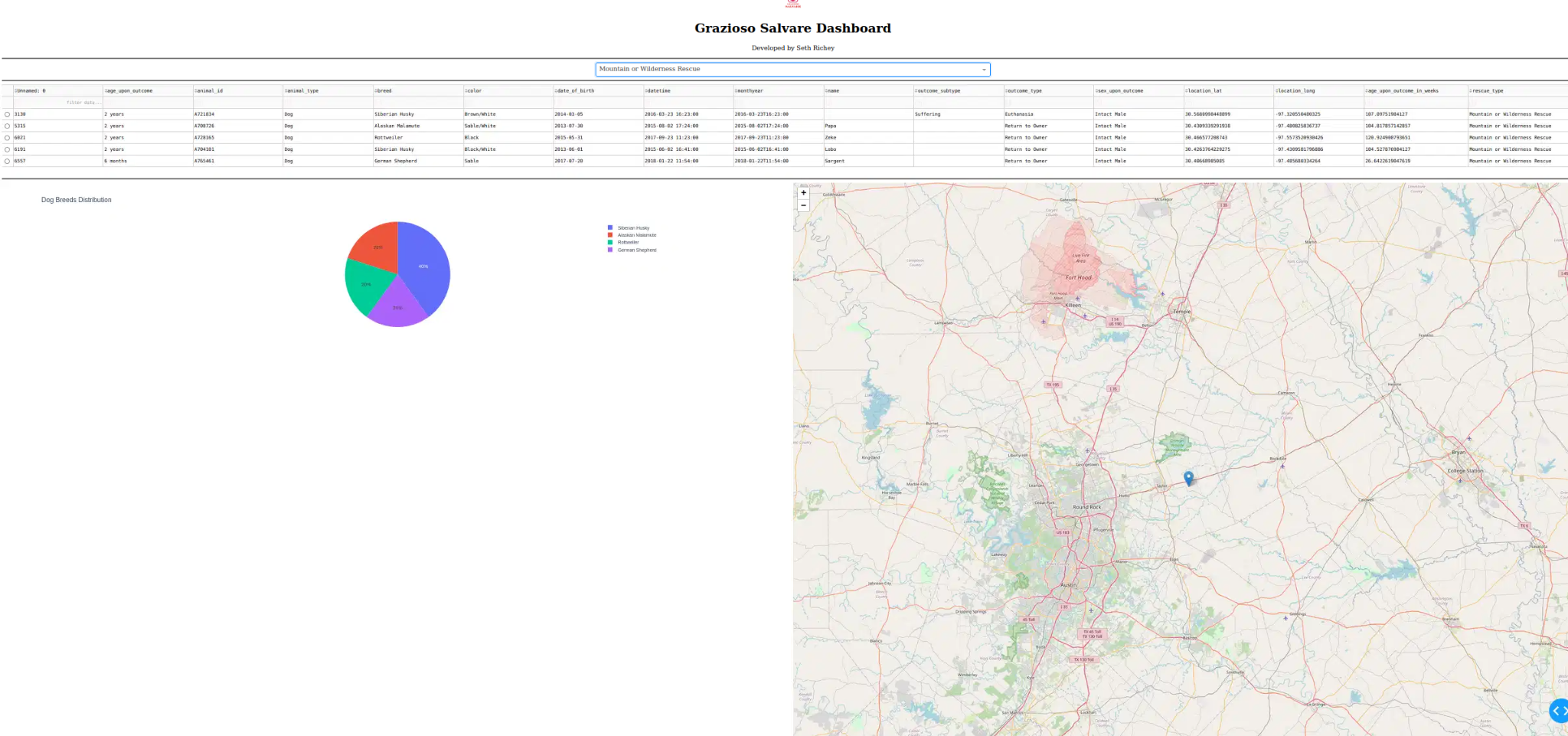
The **Grazioso Salvare Dashboard** is a web application designed to help identify dogs suitable for search-and-rescue training.

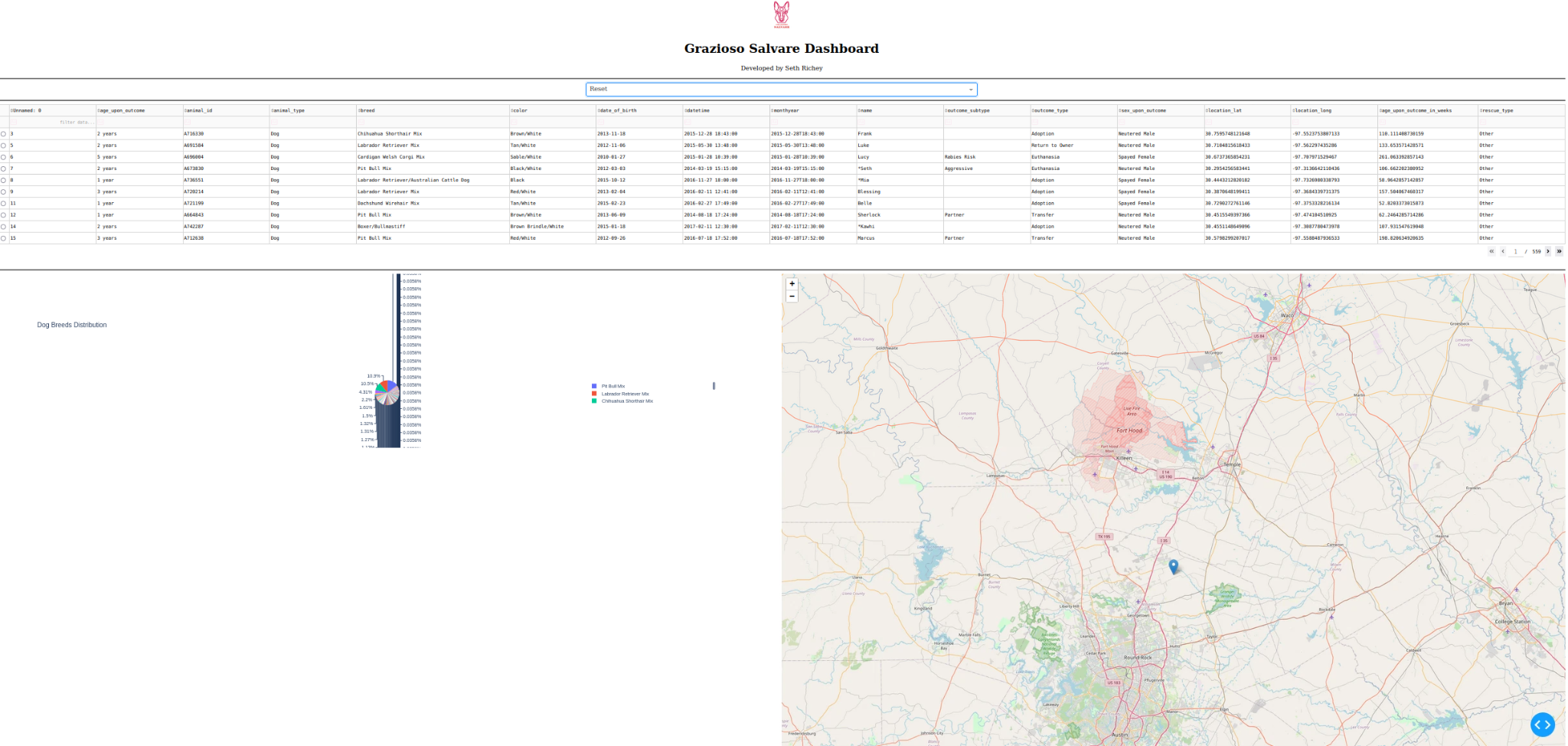
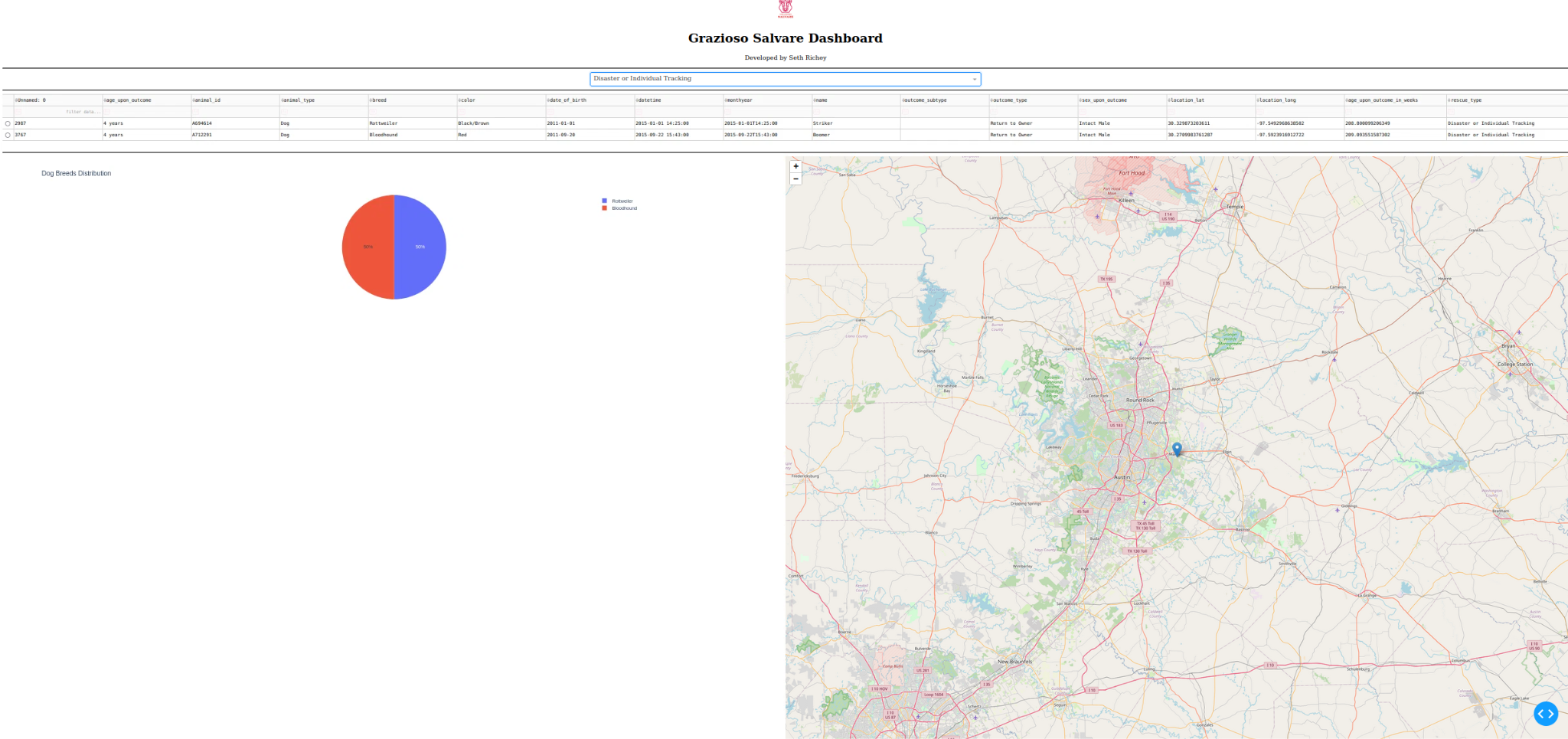
**Key Features:**

* **Interactive Filters**: Filter dogs by rescue type:  
  + Water Rescue
  + Mountain or Wilderness Rescue
  + Disaster or Individual Tracking
  + Reset (show all dogs)
* **Data Table**: Displays dog information including breed, age, sex, and outcome. Updates automatically based on the filter.
* **Charts**:  
  + Pie chart of dog breeds
  + Map showing dog locations

Screenshots:







## Tools Used:

| Tool | Purpose | Rationale |
| --- | --- | --- |
| MongoDB | Database to store dog records | Flexible document storage, easy JSON-like queries with Python |
| Dash / Plotly | Web app framework | Separates view (layout) and controller (callbacks) for dynamic dashboards |
| Pandas | Pandas | | Efficient reading, filtering, and processing of CSV data | | --- |  |  | | --- | |
| | Dash Leaflet | | --- |  |  | | --- | | | Interactive maps | | --- |  |  | | --- | | | Display geolocation of dogs on a map | | --- |  |  | | --- | |
| Dynamic, visually appealing charts | Chartsv | Dynamic, visually appealing charts |

**Why MongoDB**:

* Stores unstructured data efficiently
* Supports querying by breed, sex, and age
* Integrates easily with Python for data analysis and filtering

**Why Dash**:

* Provides view (layout) and controller (callbacks) separation
* Simplifies building interactive web dashboards entirely in Python
* Updates tables, charts, and maps dynamically when filters change

**Project Steps**

* Imported CSV data using Pandas
* Created MongoDB database to store dog records
* Built Dash layout with logo, filters, table, charts, and map
* Implemented callbacks to update all widgets based on filter selection
* Tested all filters to ensure proper functionality
* Captured screenshots for documentation

**Challenges and Solutions**

* Filtering multiple conditions: Needed to account for breed, sex, and age simultaneously
  + Solution: Created a function to assign rescue type based on Grazioso Salvare’s criteria
* Map display issues: Initial markers did not render correctly
  + Solution: Ensured correct latitude/longitude and used a default row when no selection is made
* Dynamic updates: Table, pie chart, and map needed to update seamlessly
  + Solution: Used Dash callbacks with the DataTable’s derived\_virtual\_data