**1. Project Overview & Required Functionality**

Grazioso Salvare’s goal is to identify and categorize dogs from the Austin Animal Center Outcomes dataset for search-and-rescue training. This dashboard allows users to:

1. **View all shelter dogs** in an interactive data table (with pagination and sorting).
2. **Filter by Rescue Type** using radio buttons
3. **Visualize locations** on a U.S. map (scatter-geo) of the (unfiltered) candidates.
4. **See breed distribution** in a dynamic bar chart of the (unfiltered) data.

**2. Tools & Rationale**

* **Python 3.x** & **pip** for package management.
* **MongoDB** as the database (model)
  + JSON-style documents map naturally to Python dict objects.
  + Flexible schema accommodates varied animal records.
  + Official [PyMongo driver](https://pypi.org/project/pymongo/) simplifies CRUD operations.
* **Dash by Plotly** as the web framework (view & controller)
  + Built on Flask + React—allows pure-Python callbacks to drive interactive UIs.
  + Native support for data tables, graphs, and layout.
  + Easy to deploy as a standalone or embed inline in Jupyter.
* **Plotly Express** for concise charting (scatter\_geo & bar).
* **JupyterDash** (optional) to embed inline in notebooks during development.

**3. System Requirements & Resources**

* **Python 3.8+**
* **MongoDB Server** (4.0+)
* **Node.js** (for any front-end assets, if you extend this)
* **Libraries**

**Datasets & Assets:**

* **Austin\_Animal\_Center\_Outcomes.csv (project file)**

**4. Setup & Reproduction Steps**

1. Create the database & user (in mongosh)
2. Import the CSV
3. Configure connection
4. Run the dashboard

**5. Implementation Steps**

1. **Project One**: built crud\_script.py with create/read/update/delete methods against MongoDB.
2. **Initial Notebook**: loaded raw data into a Pandas DataFrame.
3. **Column Normalization**: dropped \_id, renamed fields (e.g. rescue\_type→RescueType, location\_lat→lat, etc.).
4. **RescueType Derivation**: originally we planned to map breeds to rescue types; however, because the dataset lacks explicit rescue-type labels, or I was doing something wrong.
5. **Dash Layout**: added a spot for a logo, title, developer tag, radio-filter controls, interactive DataTable, and two charts.
6. **Callbacks**: wired filter → table + map + bar chart with a single multi-output callback (the filters currently show/hide UI but don’t alter data).