# CS 340 README Project 2

*Project Two Dashboard Functionality Screencast:*

## About the Project/Project Title

*Grazioso Salvare Full Stack Development Project* – A user-friendly software application used to identify dogs that are good candidates for search and rescue training.

This open-source project has been developed for *Grazioso Salvare* with the primary goal of facilitating user database access. The software application works with existing data from animal shelters to identify and categorize available dogs and features an easy to navigate client-facing web application dashboard. The database is managed using MongoDB, and the CRUD functionality serving as the middle-ware layer of the stacked development has been implemented using Python and the PyMongo library. The major benefit to using MongoDB and Python is that they are both dynamically typed languages. They are less strict during compile time and in effect offer developers greater options for customizations all while using less code. These tools have been chosen for their speed and flexibility during data access but also to simplify integration with the client-facing web application. The web application has been developed in Python using the pandas Python data analysis library and the Dash framework. pandas and Dash provide data manipulation flexibility and simplify the process of creating reactive, web-based applications. Using a custom layout and callback function application structure, Dash utilizes user input to dynamically respond to user requests and interactively displays these changes in the application interface. Building a project on these existing technologies offers many efficiencies including faster development times and less hassle during integration since most of the code has already been thoroughly tested.

## Motivation

*Grazioso Salvare* is an innovative international rescue-animal training company seeking qualified dogs for search and rescue training. *Grazioso Salvare* has reached an agreement with a non-profit agency that operates five animal shelters in the region around Austin, Texas and has been provided with data from their shelters. As part of their work, *Grazioso Salvare* looks for specific traits in the dogs they train. For example, different breeds of dogs are proficient at different types of rescues, such as water rescue, mountain rescue, or locating humans after a disaster so it is important to identify and categorize available dogs based on certain criteria. Utilizing open-source technologies, this user-friendly interface has been developed to facilitate these processes for users at *Grazioso Salvare*.

## Getting Started

To get a local copy of this system up and running, follow these steps:

* Before we run the web application it may be helpful to understand how the back end and middle-layer technologies are being used in this development. Please follow along with the tutorial included in this README file to learn how the MongoDB database has been created, how users have been granted access using authentication, and how the CRUD functionality operates by running the test script in Jupyter Notebook. The README file can be accessed here: 
* If the MongoDB database is operating on the local system, verify that it is currently running. Open the terminal and enter this command: **mongod\_ctl start**
* With the MongoDB database running and fully authenticated, verify that the following tools have been downloaded/installed prior to running the web application dashboard:
  + Download *ProjectTwoDashboard.ipynb* from this provided link:



* + If you followed along with the previous tutorial, you should have downloaded *AnimalShelterModule.py* from the provided link. If not, you will need to download this Python module as well:



* + - Place both the Jupyter Notebook (.ipynb) file and the Python (.py) file in the same directory.
  + <https://dash.plotly.com/installation>

Install Dash using pip. Dash will be needed to generate all client-facing features of the user interface. From this page be sure to **also** install the following:

* + - <https://github.com/plotly/jupyter-dash>

Jupyter Dash will be needed to run the web application in Jupyter Notebook.

* + - <https://pandas.pydata.org/>

Install pandas using pip. pandas will be needed to manipulate data retrieved from the MongoDB database using the PyMongo CRUD functionality. pandas is responsible for providing the data in the proper format for Dash to populate the data visualizations in the user interface.

* + - <https://plotly.com/python/plotly-express/>

Plotly Express will be used to generate the pie chart.

* + - <https://pypi.org/project/dash-leaflet/>

Dash Leaflet will be used to generate the geolocation chart.

* + - <https://numpy.org/install/sdfdsfs>

Verify that NumPy has been installed on the system. The pandas library is built on functionality provided by NumPy.

* We are now ready to launch *ProjectTwoDashboard.ipynb* using Jupyter Notebook. For an in-depth explanation of the application’s features and functionality, see the **Usage** section below.

## Installation

* <https://dash.plotly.com/installation>

Install Dash using pip.

* <https://github.com/plotly/jupyter-dash>

Jupyter Dash will be needed to run the web application in Jupyter Notebook.

* <https://pandas.pydata.org/>

Install pandas using pip.

* <https://plotly.com/python/plotly-express/>

Plotly Express will be used to generate the pie chart.

* <https://pypi.org/project/dash-leaflet/>

Dash Leaflet will be used to generate the geolocation chart.

* <https://numpy.org/install/sdfdsfs>

Verify that NumPy has been installed on the system.

* 

See this README for additional information on the back end and middle-layer technologies used in this development.

* 

This Python module will be needed to access the MongoDB database.

* 

Download the web application.

## Usage

*ProjectTwoDashboard.ipynb*

*To see the following functionality in action please view the Project Two Dashboard Functionality Screencast here:*

* Import all necessary libraries and modules.

Text

Description automatically generated

* Creating an instance of the AnimalShelter class establishes a connection to the local MongoDB database. Be sure to use the username and password for the “aacuser” account for proper authentication. After the connection is established, we utilize the CRUD Python module by running a “retrieve all” query to bring in the data from MongoDB.

*Text

Description automatically generated*

* Use Jupyter Dash and the Dash framework to start constructing the application view. Load the image supplied by the client and incorporate the logo into the display using CSS styling. If a user clicks on the logo, the user will be taken to the client’s web page: <https://www.snhu.edu>. Directly below the image logo, a unique identifier has been included to credit the developer.

*Text

Description automatically generated with medium confidence*

* Populate the view with filtering options using radio buttons. The values associated with each radio button will be passed to the callback function used to filter and update the data present in the data table, pie chart, and geolocation chart.

*A picture containing text

Description automatically generated*

* The client requested that the data table be easy to use and navigate. Custom features have been incorporated that include custom field filtering, sorting, pagination, and page size limitation.

Text

Description automatically generated

* Set up the view to show pie chart and geolocation chart side-by-side.

Graphical user interface, text

Description automatically generated

* This callback function filters data as specified by the client. When a radio button is selected on the interface, the input is passed to the function and a new data frame is generated consisting of the data specific to the filter request. If a user needs to reset the data to the initial state, the user can select the Reset radio button option. The data frame and columns are returned to the application layout as well as the current page number with a value of zero. When swapping filters that generate multiple pages of data to filters that generate only a single page of data, a bug was found that would not show any data for the filter requiring only a single page. By resetting the page number to zero, the filters and interactive updates work and appear as they should.

Text

Description automatically generated 

* Basic callback to highlight a selected column.

Text

Description automatically generated with medium confidence

* Use pandas and Plotly Express to generate pie chart consisting of breed ratios. This callback responds to changes made in the data table.

Text

Description automatically generated

* Use pandas and Dash Leaflet to generate geolocation chart showing the first animal in the data table’s location. The map generates a marker on the geolocation map relative to the first animal’s latitude and longitude location and will respond to changes made in the data table. A tooltip shows the animal’s breed and when the marker is clicked a popup displays the animal’s name.

Text

Description automatically generated

* Run the application.



## Contact

Scott Smith

scott.smith15@snhu.edu