<u>Developer Documentation</u>

Environment Setup:

(Note: this assumes you already have Python up and running on your machine.)
1. To ensure smooth setup and avoid conflict with other libraries, we recommend setting up a virtual environment.
    a. A word of warning: virtual environments created in an Ubuntu/WSL terminal can't be accessed/run from the terminals built into VSCode and vice versa. If you are doing your coding in VSCode, we *highly* recommend that you create your virtual environment *in VSCode.*
    b. Directions to create a virtual environment
        i. Mac/linux system
            1. Make sure you have python installed and the latest version being python3. You can do python3 --version to check your python version
            2. Create a virtual environment
                a. Python3 –m venv <environment name>
                b. You can just do python3 –m venv venv
            3. Activate the virtual environment
                a. Source venv/bin/activate
                b. Install all packages within the virtual environment by doing pip install
            4.
    c. [Resources](#)
2. Once you have your virtual environment up and currently running, use the command <pip install -r requirements.txt> in your terminal to install all the supplementary libraries.

The Important Files Where Most of the Program Is Contained:

a. Main.py - This is the main script. It calls other functions and integrates all the functionalities into one place where everything can be run.
b. GUIexpirmentation.py - This contains all the code that creates the GUI and handles its events.
c. rows.py - This is where the functions to add and remove columns and rows reside. We were still in the process of attempting to integrate it into the GUI when we ran out of time, so...
    a. In the main branch on GitHub, it is designed to work in the terminal and is not connected to the GUI at all yet.
    b. The frazfinal branch is where the incomplete attempt to integrate rows.py into the GUI currently resides. It is not complete, and thus crashes fatally when run.

d. read.py - This contains all the functions for searching for data in the Excel file and returning the cell that you are looking for.
e. openpyxl – This is the folder containing all the code in the openpyxl supplementary library.
f. FoundHouse.xlsx - This is the training Excel workbook, containing the fake data we were given so we could test our program. Its file path is hardcoded into our code, so that might need to be changed down the line.
g. Attribute.py, Owner.py, and Pet.py - These are leftover files from when we were considering an object-oriented approach. We're keeping them here just in case they're useful down the line.
   a. Attribute objects were going to be a specific property of each pet – analogous to the columns in the Excel sheet.
   b. Pet objects were going to be collections of information about each pet – analogous to the rows in the Excel sheet.
   c. Owner objects were going to be collections of information about each owner - similar to Pet objects, but they were going to be distinct, as Pets and Owners would have different Attributes.

User Documentation/Features:

1. Main Features
   a. Searching for values
      i. Enter a search term, and the application will look through the Excel file. The results will be displayed in the box on the left with the specific cell location. Please note that we do not use spaces, it may stop the code from reading the value correctly.

      ii.
         1. To filter by single parameters value
            a. Type the values you want to filter by(comma separated) for multiple inputs
               i. Example: Cat and Dog
            b. The result will appear on the right and it should give the cell location that contains cat and dog, including their associated values such as the name of the owner, etc...

    iii.

1. To filter by conditions
   a. Enter the column name
   b. Use any condition operations and the value you want to check
   c. The goal was to make it allow you to filter for every Dog that has an age of less than a certain number(ex: Type=Dog,Age<6)


    iv.

1. To search for a single value
   a. Select the sheet you want from the drop-down menu
   b. Enter the name of the column (it can read the column headers)
   c. Enter the target in the search field
   d. The result will be displayed in the right panel with the cell location of the target, and it does not matter how you search for the target.
2.

    v.

b. Count the animals in the shelter
   i. Click the "Count Animals" button and it will display the number of animals in the shelter and how many of each type of animal
   ii. Example:
      1. Total number: 10
      2. Dogs; 5
      3. Cats: 3
      4. Guinea pig: 2
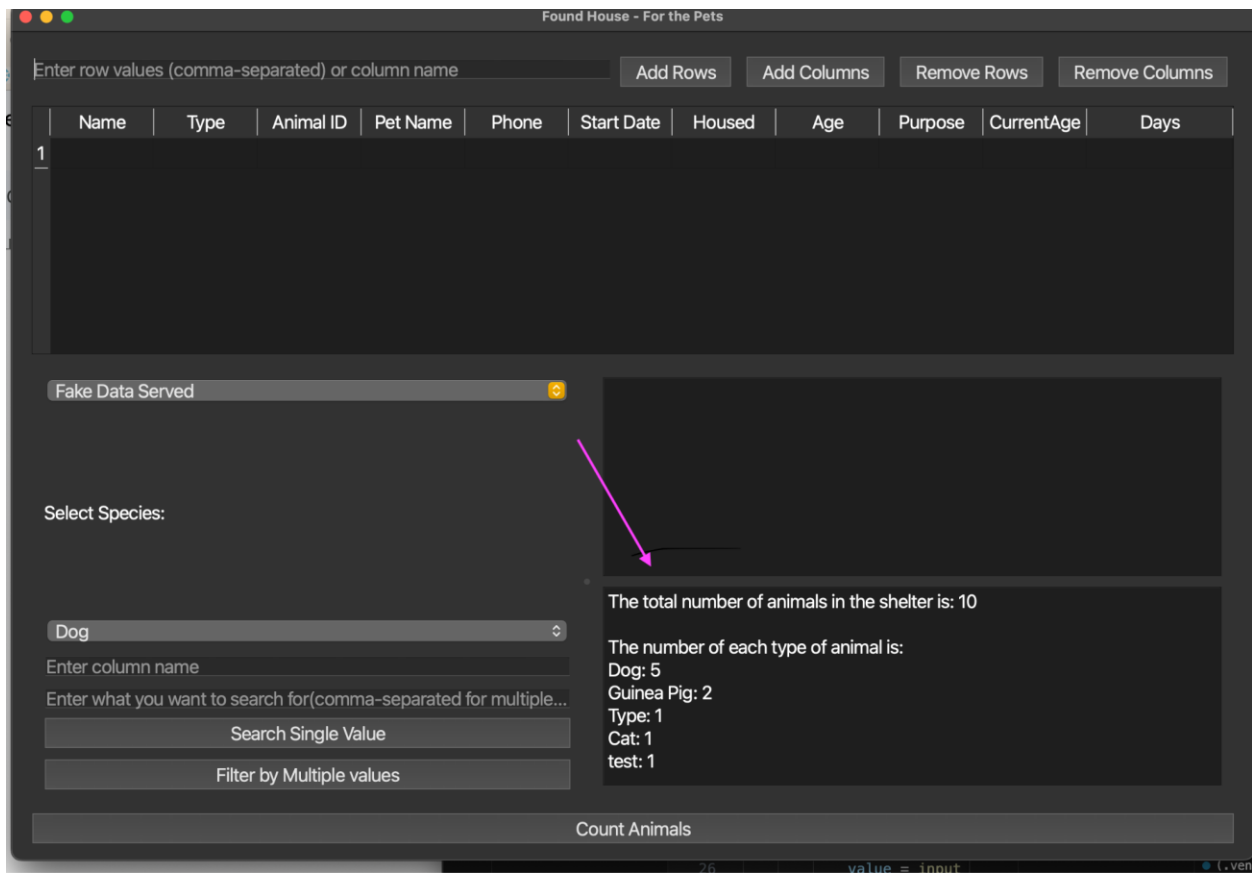
2. Adding and Removing Columns and Rows
   a. There is code for adding and removing columns and rows, but it is not connected to the GUI yet. Once it is connected it will ask the user to input a value for each cell and then it will save that to the excel sheet so you are able to search for the values later on if needed
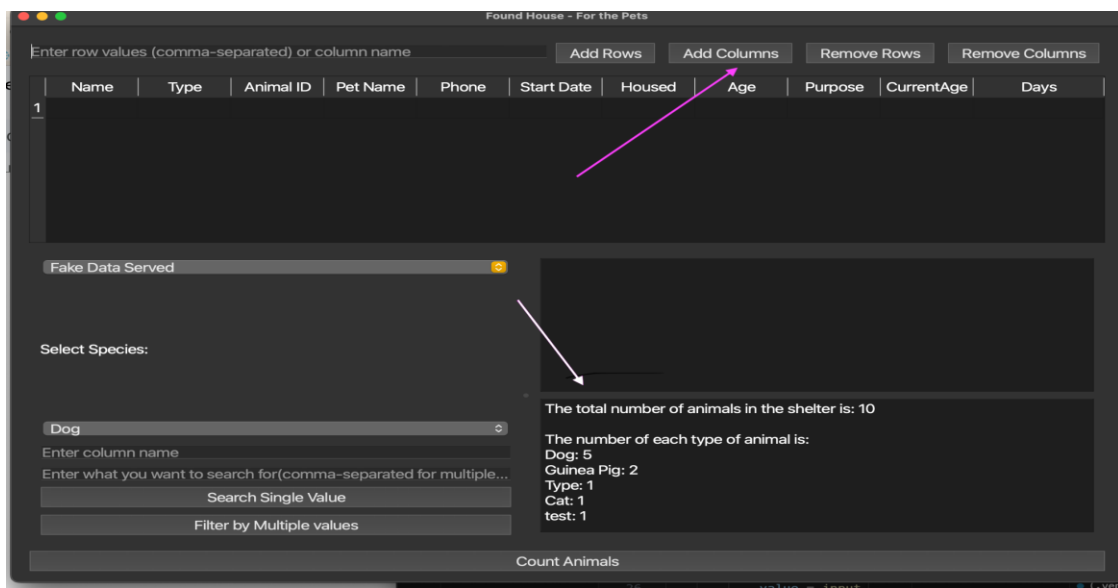
B. You can also select which sheet in the excel file you would like to work on by clicking on the current sheet name, in this case it is "Fake Data Served".