Tomislav Simic

Professor: D. Marrero

Class: 12P

Date: 2024-02-29

Name of the application: FurBabyBoarding

Overview

Keep track of available boarding spots based on actual events vs daily spot allocation. No real pet or owner information is gathered due to HIPPA-like vet patient data protection.

Reason

The current application in ‘use’ is being avoided as the tracking is based on static date entries, with disregard to actual occupancy at any given time. Current workaround implemented is sticky notes that don’t have a proper tracking and convoluted process that is difficult to train new hires on.

Target Audience

* Staff working with animals:
  + Young man and women, in their 20’s with completed high school and some college.
* Animal hospital management (if the application is found feasible):
  + Different age and gender doctors and managers

Objectives:

*Prompt 1*

Allow users to enter pet name and assign a specific type of kennel. This step has been completed.

*Prompt 2*

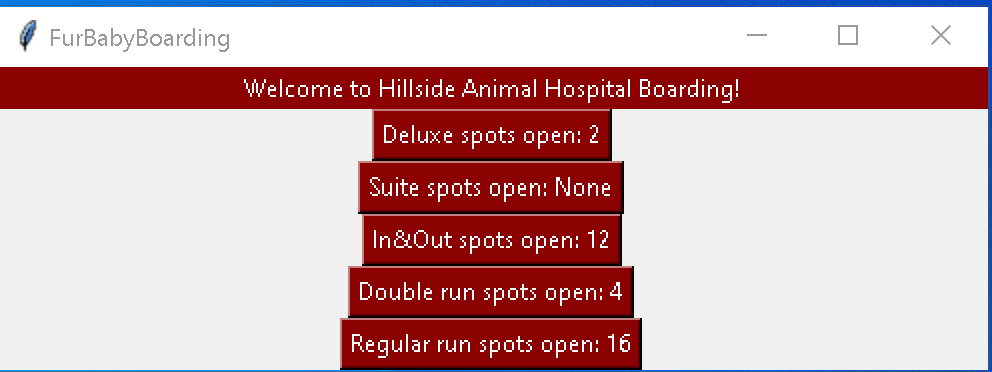
Calculate open spots based on live data and display all available spot and their quantities with 2-week look ahead.

*Prompt 3*

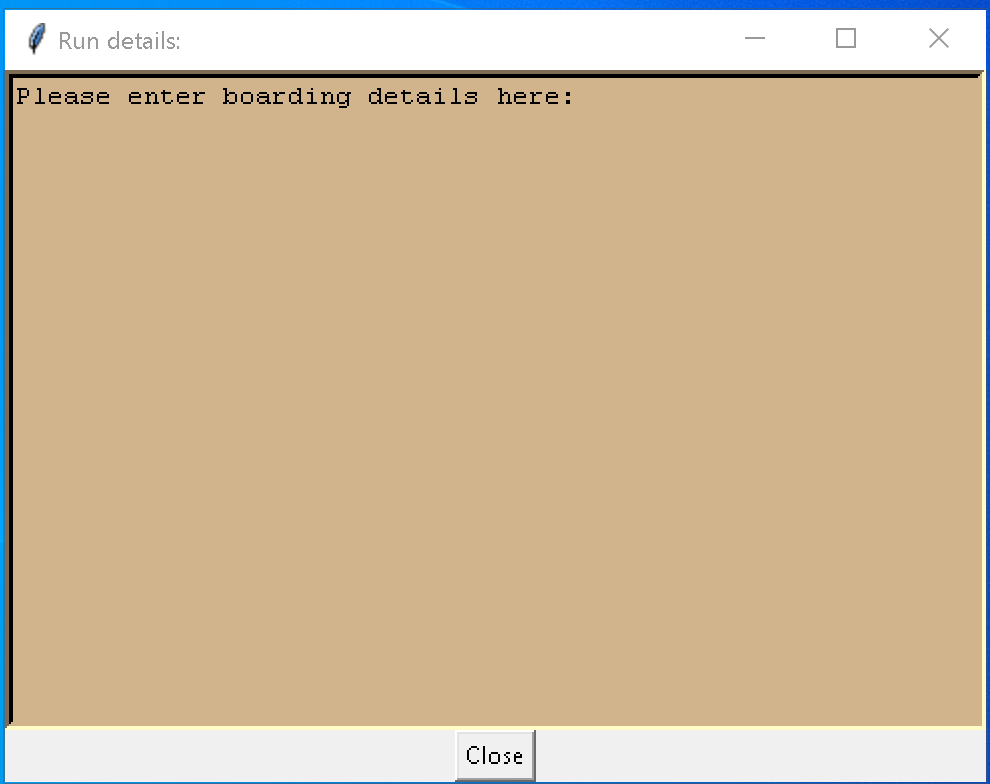
Display a warning if attempting to book a spot that has been already filled. Otherwise, prompt for user to enter basic pet information

Sample Wire Screen

*Window 1*



*Window 2*



Types of kennels and their quantities:

* 2 -3 pets:
  + Deluxe = 2
  + Suites = 7
  + In/Out :
    - Front = 7
    - Back = 5
  + Regular = 16
* Single pet:
  + Cages:
    - Large = 2
    - Medium = 4
    - Cat = 7
    - Rabit = 2
  + Stalls = 2

Current State:

I still have to create functions to calculate current boarders vs incoming, which will react with either providing the input window or show warning and ask to proceed.

Still need to create a function to handle database entries.

I’m open to suggestions if anyone knows a better way to put the classes together instead of random class for the 2nd pop-up window but anything I have tried so far, either messes up rest of the code or is littered with errors and PEP8 violations.

Updates made until 3-5-2024:

Added functions:

* Fielded -> Creates set of fields to represent available runs instead of showing them within the button.Updates made today:

Added classes:

* User input verification test unit -> checks user’s input against correct format and shows corresponding messages, and returns tuple of date segments

Updates made 3-7-2024:

Added functions:

* Reaction -> Implements changes to the display field based on logic and tested it
* Spots\_available -> Checks the suites in dictionary for availability and updates the information based on user’s selection

Additional changes:

* Moved the READme.txt file to new folder (FurryResources)
* Modified code to reflect the READme.txt location
* Added picture for testing as additional label for the requirements
* Added icon to all the windows
* Made the App class UserInput’s class subclass
* Added label to entry window for user’s visibility
* Collapsed 2 functions for help window into a single get\_help function
* Added another label to show a picture
* Changing the name of this document to “Final\_Project\_Process\_documentation”

Finalization update:

* Added second picture within the help window
* Reconfigured all buttons to follow correct logic
* Created additional functions within Reaction function:
  + Clear\_entry => provides a means to clear entry/display field
  + Processing\_input => provides a interface to UserInput Class’ date\_format function, provides logic to display messages to user, and calls on entry widget when applicable.
* Modified the prompts dictionary to include message for empty input after migrating the directory from an isolated class into the App class
* Swapped pictures within the GUI
* Cleared up pseudo code and updated in-code documentation
* Rearranged functions to follow logical use.
* Added empty entry verification to the date\_format function in the UserInput class
* Modified processing\_input function to display date entered, after verification in user friendly format

Content of the READme.txt file:

Program: FurBabyBoarding.py

Author: Tomi Simic

Last date modified: 2024-03-09

Use case:

This program is used to keep track of available runs inside an animal boarding

facility. The GUI is providing the means to gather user selection while

the verification algorithm checks user input for typos and logical errors.

Instructions:

The program is providing an interface that includes 10 dark red buttons for selection of type of run you would like to utilize for the boarding pet. Right beside the buttons is the field depicting the run availability, respectively. Aside from the Welcoming label in read, with the inscription “Welcome to FurBaby Boarding!”, you have the main display with the same inscription that also functions as an entry field.

The steps you should take are as follows:

1. Click on the type of run to be utilized.
2. This will show the change of available runs in the field adjacent to the button clicked. Also, the ‘Main Display’ field will now show instructions to be followed.
3. Click on the ‘Main Display’
4. Enter the day on which the pet will be leaving our boarding facility
5. Click on ‘Enter’ button. Please note that the button will remain disabled until you click on the run selection button!
6. Please utilize the date format: mm-dd-yyyy
7. If you make a mistake, the display field will prompt you with the instructions to be taken to correct it.
8. Once the correct date has been entered, you will be prompted to enter pet information and you will be provided with current date and time stamp that will be logged at a later time, automatically.
9. Click on ‘Save’ button to exit the screen and return to main interface for next pet boarding.

Testing Unit:

* Cleaned up the TestUnit.py utilized to test the user input verification
* Used following inputs to test the code:
  + 45/45/45
  + Hoh-45-
  + Hoh-45-1245
  + 12-32-1236
  + 3-16-1245
  + 3-16-2024 (to verify tuple return)
  + Empty entry
* Manually tested the GUI
  + by entering same values utilized in TestUnit.py for input verification
  + by clicking on ‘Return’ button for the display/input field
  + by clicking on buttons for each run, respectively
  + verified that logic follows up to the pet data input window through the ‘Save’ button which is still just closing but the logic is in place to replace that command with future command to save the input into database
  + verified that ‘User Manual’ button brings up the READme.txt file

Testing results of the GUI:

