Cole Bodine

Project: Weather Data Analysis System

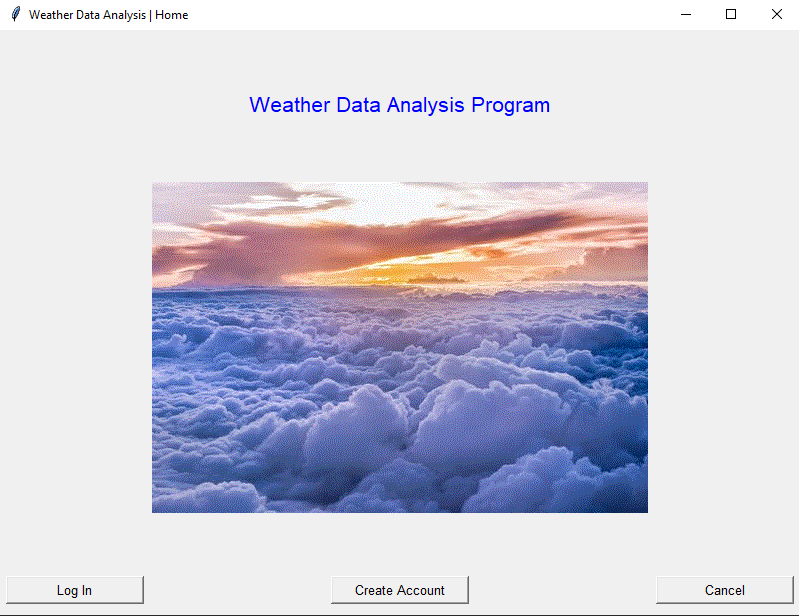
Design Document

CSE-222-101

This project is a program that allows a user to log in and access a data analysis system. The data analysis system extracts data from a data library file and allows the user to select a data set and time period to be plotted.

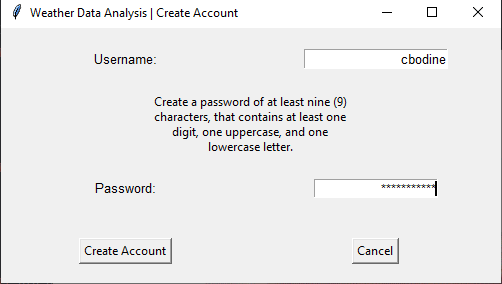
**Project Milestone #1 – Main GUI, Account Creation, and Password Handling**

When the program is initially started, the user will be taken to the screen shown in Figure 1. The screen contains user login, account creation, and exit buttons.



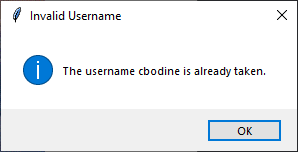
*Figure 1: Home GUI*

If the user does not have an account, they can create one by clicking the “Create Account” button shown in Figure 1. Upon clicking the button, they will be taken to a new window, where they can enter their username and password, as shown in Figure 2.

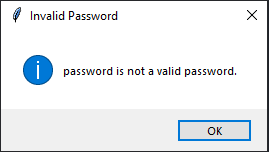


*Figure 2: Account Creation Window*

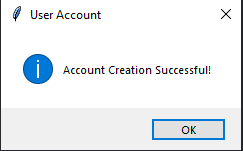
If the user enters a username that has already been taken, they will receive an error telling them that the username has been taken (Figure 3). Likewise, if they enter an invalid password, they will receive a prompt informing them that the password is not valid (Figure 4). Finally, if the user enters a valid username and password, their account will be created and stored in a file (Figure 5).



*Figure 3: Invalid Username*

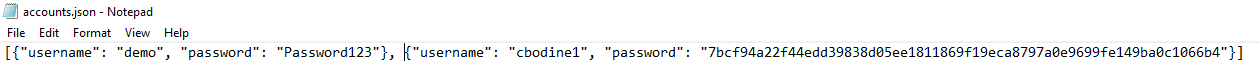


*Figure 4: Invalid Password*



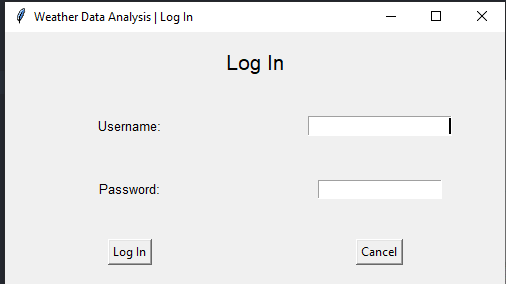
*Figure 5: Successful Account Creation*

When an account is successfully created, it is appended to a list of dictionaries in a file called “accounts.json” (Figure 6) that resides in the same location as the Python program. Before being stored, the password is passed through a hashing function provided by the hashlib library, which encrypts the password.



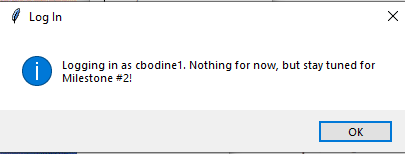
*Figure 6: Username and Password as stored in the file.*

After the user’s account is created, the account creation button is disabled, and the login button is re-enabled. If they could not already, the user may now log in by cligking the “Log In” button from Figure 1. This will take them to the screen shown below, in Figure 7.



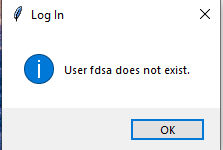
*Figure 7: Login Window*

Like with the account creation window, activating the login window deactivates the “Log In” and “Create Account” buttons from the main window. Here, the user will be prompted to enter a valid username and password. The program determines which username and password is valid by opening the “accounts.json” file and parsing through the user\_accounts list. If the program finds a dictionary that contains a username and password matching what the user entered, that user will be logged in to the system (Figure 8). This doesn’t do anything besides display a message right now.

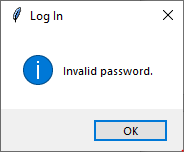


*Figure 8: Successful Login*

If they provide an invalid username (Figure 9) or password (Figure 10), they will receive a prompt informing them.



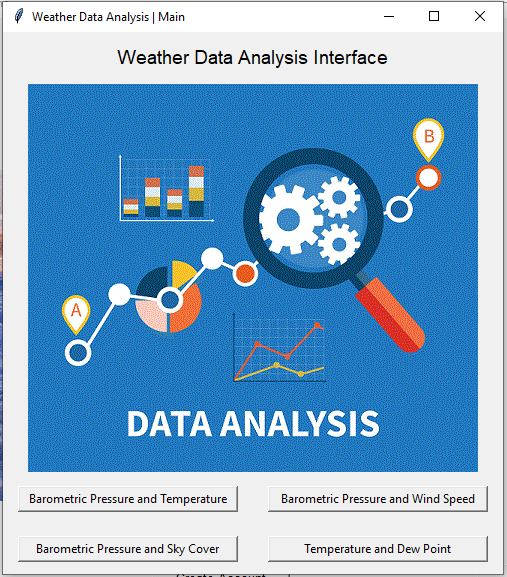
*Figure 9: Invalid User*



*Figure 10: Invalid Password*

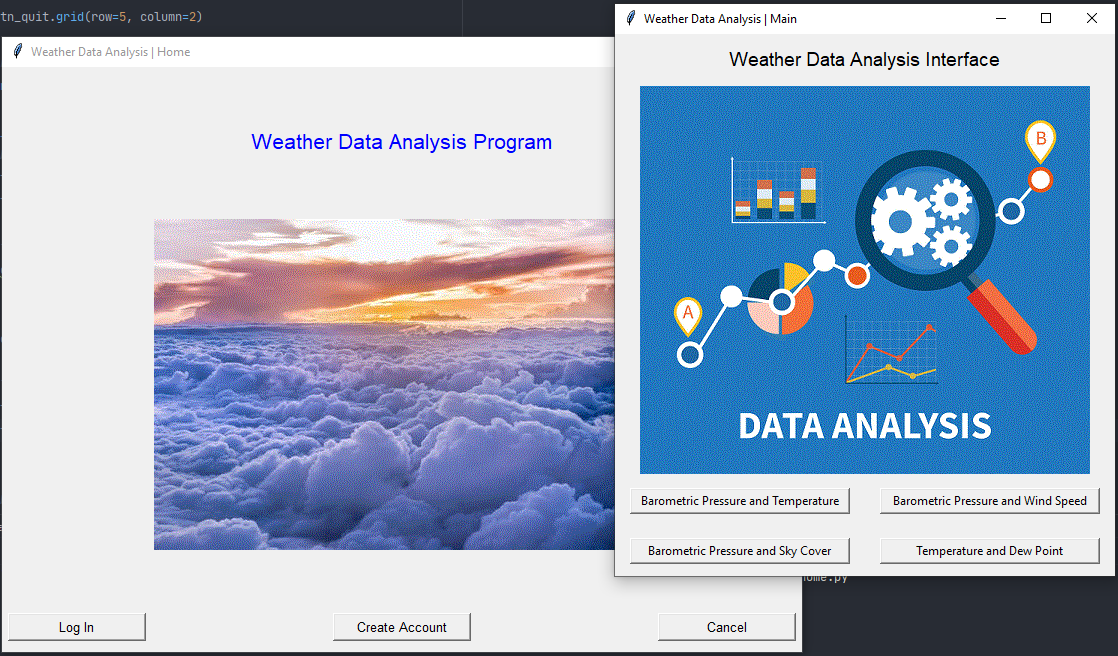
**Project Milestone #2**

The program now has complete login functionality and a main GUI that will allow the user to select which data they would like to view. The message box from Figure 8 has been removed and logging in will now take the user to the main GUI, as shown in Figure 11.



*Figure 11: Main GUI.*

Most of the other work done to the project is simple maintenance and changing the code around, as I accidentally went too far in Milestone #1 by completing the login process that was meant to be saved for Milestone #2. For example, I renamed the original main.py to home.py and changed any code with the word main to have the word home, to make it more easily differentiable from the new main.py. I opted for creating an entirely new window rather than morphing the original because I want the user to still have access to the home window in case they want to log out of the program and let someone else use it without having to close the entire program. Now, all they will have to do is close the main window and they can start the program anew by logging in as someone else (Figure 12).



*Figure 12: Allow the user to log in as someone else after closing main interface.*

**Project Milestone #3**

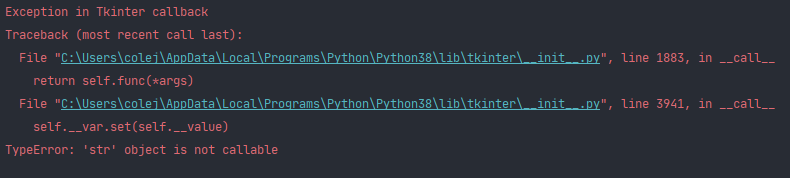
Milestone #3 has proven to be more difficult than I expected. The basic idea of what I was trying to do was for the user to be able to click on one of the options on the main GUI, which would then take them to a window that would provide them with two dropdown menus, allowing them to pick a range of dates like the diagram shown in Figure 13.

Graphical user interface, text, application

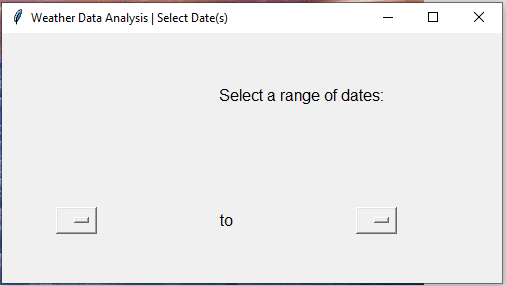
Description automatically generated

*Figure 13: Mockup design of Window*

Unfortunately, I ran into some issues that did not allow me to complete the Milestone completely. The main problem I ran into with the code was that my variable for the list of dates was not being passed correctly to the tkinter OptionMenu (the dropdown menu). This left me with the error code seen in Figure 14 and the GUI shown in Figure 15.



*Figure 14: Error code*



*Figure 15: The broken GUI*

The callback error comes from the tkinter library’s \_\_init()\_\_ function. For whatever reason, my list of dates looks like it is being passed as a string, rather than a list. My idea to fix this problem between now and Milestone #4 is to break this problem down into a smaller size by isolating it in its own environment (to give me less things to work with). Once I figure it out in a smaller setting, I will adapt that solution to the larger setting that is this project.

**Current State of Code**

# home.py

# This file stores the home window for the weather data analysis project.

from tkinter import \*

import tkinter as tk

import create

import login

import main

# home window

class HomeGUI:

"""A class holding the home window."""

def \_\_init\_\_(self):

"""Initialize the home window."""

# Create the home window.

self.win\_home = tk.Tk()

self.win\_home.title("Weather Data Analysis | Home")

self.win\_home.minsize(width=450, height=300)

# Configure columns

self.win\_home.columnconfigure(0, minsize=150)

self.win\_home.columnconfigure(1, minsize=150)

self.win\_home.columnconfigure(2, minsize=150)

# Configure Rows

self.win\_home.rowconfigure(0, minsize=50)

self.win\_home.rowconfigure(1, minsize=50)

self.win\_home.rowconfigure(2, minsize=50)

self.win\_home.rowconfigure(3, minsize=50)

self.win\_home.rowconfigure(4, minsize=50)

self.win\_home.rowconfigure(5, minsize=50)

# Create label widget.

self.lbl\_header = tk.Label(text="Weather Data Analysis Program",

font=("Helvetica", 16),

fg="blue")

self.lbl\_header.grid(row=1, column=0, columnspan=3)

# Create the picture widget.

photo = PhotoImage(file="weather.gif")

self.labelGIF = tk.Label(image=photo)

self.labelGIF.image = photo

self.labelGIF.grid(row=3, column=1)

# Create the button widgets.

self.btn\_login = tk.Button(text="Log In",

font=("Helvetica", 10),

width=16,

command=self.log\_in)

self.btn\_create\_acct = tk.Button(text="Create Account",

font=("Helvetica", 10),

width=16,

command=self.create\_account)

self.btn\_quit = tk.Button(text="Cancel",

font=("Helvetica", 10),

width=16,

command=self.win\_home.destroy)

self.btn\_login.grid(row=5, column=0)

self.btn\_create\_acct.grid(row=5, column=1)

self.btn\_quit.grid(row=5, column=2)

# Enter home tkinter loop

tk.mainloop()

def create\_account(self):

# Disable the buttons

self.btn\_create\_acct.config(state=DISABLED)

self.btn\_login.config(state=DISABLED)

# Create an account creation GUI

self.acct\_GUI = create.CreateGUI()

# Wait for the window to be destroyed.

self.acct\_GUI.win\_create.wait\_window()

# Enable login button again.

self.btn\_login.config(state=NORMAL)

self.btn\_create\_acct.config(state=NORMAL)

def log\_in(self):

# Disable the buttons

self.btn\_create\_acct.config(state=DISABLED)

self.btn\_login.config(state=DISABLED)

# Create an account creation GUI

self.login\_GUI = login.LoginGUI()

# Wait for the windows to be destroyed.

self.login\_GUI.win\_login.wait\_window()

# Enable login and create buttons again.

self.btn\_login.config(state=NORMAL)

self.btn\_create\_acct.config(state=NORMAL)

dataProgram = HomeGUI()

################################################################################

# create.py

# This file holds the account creation window and logic.

from tkinter import \*

from tkinter import messagebox

import tkinter as tk

import json

import os

import hashlib

class CreateGUI:

"""A class containing the create account GUI"""

def \_\_init\_\_(self):

"""Initialize the AccountGUI class."""

# Create window

self.win\_create = tk.Tk()

self.win\_create.title("Weather Data Analysis | Create Account")

self.win\_create.minsize(width=500, height=250)

# Configure columns

self.win\_create.columnconfigure(0, minsize=250)

self.win\_create.columnconfigure(1, minsize=250)

# Configure Rows

self.win\_create.rowconfigure(0, minsize=62.5)

self.win\_create.rowconfigure(1, minsize=62.5)

self.win\_create.rowconfigure(2, minsize=62.5)

self.win\_create.rowconfigure(3, minsize=62.5)

# Create username widgets.

self.win\_create.lbl\_username = tk.Label(self.win\_create,

text="Username:",

font=("Helvetica", 10))

self.win\_create.lbl\_username.grid(row=0, column=0)

self.win\_create.entry\_username = tk.Entry(self.win\_create,

justify="right",

font=("Helvetica", 10))

self.win\_create.entry\_username.grid(row=0, column=1)

self.win\_create.entry\_username.focus\_force()

# Create the password widgets.

self.win\_create.lbl\_password\_guide = tk.Label(self.win\_create,

text="Create a password "

"of at least nine "

"(9) characters, "

"that contains at "

"least one digit, "

"one uppercase, "

"and one lowercase "

"letter.",

wraplength=200)

self.win\_create.lbl\_password\_guide.grid(row=1, column=0, columnspan=2)

self.win\_create.lbl\_password = tk.Label(self.win\_create,

text="Password:",

font=("Helvetica", 10))

self.win\_create.lbl\_password.grid(row=2, column=0)

self.win\_create.entry\_password = tk.Entry(self.win\_create,

width=20,

justify="right",

show="\*")

self.win\_create.entry\_password.grid(row=2, column=1)

self.win\_create.btn\_create = tk.Button(self.win\_create,

text="Create Account",

command=self.create\_account)

self.win\_create.btn\_create.grid(row=3, column=0)

self.win\_create.btn\_cancel = tk.Button(self.win\_create,

text="Cancel",

command=self.win\_create.destroy)

self.win\_create.btn\_cancel.grid(row=3, column=1)

# Lift to top

self.win\_create.lift()

def create\_account(self):

"""Create a user account."""

password = self.win\_create.entry\_password.get()

username = self.win\_create.entry\_username.get()

# If a file does not exist for user accounts, create one with

# placeholder data.

if not os.path.isfile("accounts.json"):

acct\_file = open("accounts.json", "w")

json.dump([{"username": "demo", "password": "Password123"}],

acct\_file)

acct\_file.close()

try:

acct\_file = open("accounts.json", "r")

user\_accounts = json.load(acct\_file)

except FileNotFoundError:

print(f"File {acct\_file} does not exist.")

def validate\_username(username):

"""Check to see if the username is taken."""

if not any(user['username'] == username.lower() for

user in user\_accounts):

return True

else:

tk.messagebox.showinfo("Invalid Username",

f"The username {username} is already "

f"taken.")

def validate\_password(password):

"""Validate user's password."""

long\_enough = False

has\_lower = False

has\_upper = False

has\_digit = False

if len(password) >= 9:

long\_enough = True

for ch in password:

if ch.islower():

has\_lower = True

if ch.isupper():

has\_upper = True

if ch.isdigit():

has\_digit = True

if long\_enough and has\_lower and has\_upper and has\_digit:

return True

else:

tk.messagebox.showinfo("Invalid Password", f"{password} is "

f"not a valid "

f"password.")

if validate\_username(username) and validate\_password(password):

hashed\_password = hashlib.sha256(str.encode(password)).hexdigest()

user\_accounts.append({'username': username.lower(),

'password': hashed\_password})

tk.messagebox.showinfo("User Account", "Account Creation "

"Successful!")

acct\_file.close()

acct\_file = open("accounts.json", 'w')

json.dump(user\_accounts, acct\_file)

acct\_file.close()

self.win\_create.entry\_username.delete(0, END)

self.win\_create.entry\_password.delete(0, END)

self.win\_create.destroy()

else:

print("Couldn't create account. Please try again.")

self.win\_create.entry\_username.delete(0, END)

self.win\_create.entry\_password.delete(0, END)

################################################################################

# login.py

# This login window and logic.

from tkinter import \*

import tkinter as tk

from tkinter import messagebox

import json

import hashlib

import main

class LoginGUI:

"""A class holding the login GUI."""

def \_\_init\_\_(self):

"""Initialize the login GUI."""

# Create window

self.win\_login = tk.Tk()

self.win\_login.title("Weather Data Analysis | Log In")

self.win\_login.minsize(width=500, height=250)

# Configure columns

self.win\_login.columnconfigure(0, minsize=250)

self.win\_login.columnconfigure(1, minsize=250)

# Configure Rows

self.win\_login.rowconfigure(0, minsize=62.5)

self.win\_login.rowconfigure(1, minsize=62.5)

self.win\_login.rowconfigure(2, minsize=62.5)

self.win\_login.rowconfigure(3, minsize=62.5)

# Create Login Label

self.win\_login.lbl\_title = tk.Label(self.win\_login,

text="Log In",

font=("Arial", 16))

self.win\_login.lbl\_title.grid(row=0, column=0, columnspan=2)

# Create username widgets.

self.win\_login.lbl\_username = tk.Label(self.win\_login,

text="Username:",

font=("Helvetica", 10))

self.win\_login.lbl\_username.grid(row=1, column=0)

self.win\_login.entry\_username = tk.Entry(self.win\_login,

justify="right",

font=("Helvetica", 10))

self.win\_login.entry\_username.grid(row=1, column=1)

self.win\_login.entry\_username.focus\_force()

# Create the password widgets.

self.win\_login.lbl\_password = tk.Label(self.win\_login,

text="Password:",

font=("Helvetica", 10))

self.win\_login.lbl\_password.grid(row=2, column=0)

self.win\_login.entry\_password = tk.Entry(self.win\_login,

width=20,

justify="right",

show="\*")

self.win\_login.entry\_password.grid(row=2, column=1)

self.win\_login.btn\_create = tk.Button(self.win\_login,

text="Log In",

command=self.log\_in)

self.win\_login.btn\_create.grid(row=3, column=0)

self.win\_login.btn\_cancel = tk.Button(self.win\_login,

text="Cancel",

command=self.win\_login.destroy)

self.win\_login.btn\_cancel.grid(row=3, column=1)

# Lift to top

self.win\_login.lift()

def log\_in(self):

"""Log the user in."""

username = self.win\_login.entry\_username.get()

password = self.win\_login.entry\_password.get()

hashed\_password = hashlib.sha256(str.encode(password)).hexdigest()

# Try to open the file

try:

acct\_file = open("accounts.json", "r")

user\_accounts = json.load(acct\_file)

except IOError:

tk.messagebox.showinfo("FILE ERROR", f"File {acct\_file} does not "

f"exist.")

if any(user['username'] == username.lower() and

user['password'] == hashed\_password for user in user\_accounts):

self.main\_GUI = main.MainGUI()

self.win\_login.destroy()

if not any(user['username'] == username.lower() for

user in user\_accounts):

tk.messagebox.showinfo("Log In", f"User {username} does not exist.")

self.win\_login.entry\_username.delete(0, END)

self.win\_login.entry\_password.delete(0, END)

elif any(user['username'] == username.lower() and

not user['password'] == hashed\_password for

user in user\_accounts):

tk.messagebox.showinfo("Log In", "Invalid password.")

self.win\_login.entry\_username.delete(0, END)

self.win\_login.entry\_password.delete(0, END)

################################################################################

# main.py

# This file holds the Main GUI that accesses the weather station data.

from tkinter import \*

import tkinter as tk

import select

class MainGUI:

"""A class holding the login GUI."""

def \_\_init\_\_(self):

"""Initialize the login GUI."""

# Create window

self.win\_main = tk.Toplevel()

self.win\_main.title("Weather Data Analysis | Main")

self.win\_main.minsize(width=500, height=250)

# Configure columns

self.win\_main.columnconfigure(0, minsize=250)

self.win\_main.columnconfigure(1, minsize=250)

# Configure rows

self.win\_main.rowconfigure(0, minsize=50)

self.win\_main.rowconfigure(1, minsize=50)

self.win\_main.rowconfigure(2, minsize=50)

self.win\_main.rowconfigure(3, minsize=50)

# Create Main Label

self.win\_main.lbl\_main = tk.Label(self.win\_main,

text="Weather Data Analysis "

"Interface",

font=("Helvetica", 14))

self.win\_main.lbl\_main.grid(row=0, column=0, columnspan=2)

# Create picture widget

photo = PhotoImage(file="data\_analysis.gif")

self.labelGIF = tk.Label(self.win\_main, image=photo)

self.labelGIF.image = photo # Retain a reference

self.labelGIF.grid(row=1, column=0, columnspan=2)

# Create Buttons

self.win\_main.btn\_1 = tk.Button(self.win\_main,

text="Barometric Pressure and "

"Temperature",

width=30,

command=self.open\_selection\_gui)

self.win\_main.btn\_1.grid(row=2, column=0)

self.win\_main.btn\_2 = tk.Button(self.win\_main,

text="Barometric Pressure and Wind "

"Speed",

width=30,

command=self.open\_selection\_gui)

self.win\_main.btn\_2.grid(row=2, column=1)

self.win\_main.btn\_3 = tk.Button(self.win\_main,

text="Barometric Pressure and Sky "

"Cover",

width=30,

command=self.open\_selection\_gui)

self.win\_main.btn\_3.grid(row=3, column=0)

self.win\_main.btn\_4 = tk.Button(self.win\_main,

text="Temperature and Dew Point",

width=30,

command=self.open\_selection\_gui)

self.win\_main.btn\_4.grid(row=3, column=1)

def open\_selection\_gui(self):

"""This function opens up the selection gui."""

self.select\_GUI = select.SelectGUI()

################################################################################

# select.py

# This file allows the user to select from the range of dates what they would

# like to view.

from tkinter import \*

import tkinter as tk

class SelectGUI:

"""This class holds the window for selecting a date range."""

def \_\_init\_\_(self):

"""Initialize the login GUI."""

# create variables for file, list of dates, and a temporary variable for

# the current date that will be used in the loop and logical statements.

filename = 'Data\_2010\_thru\_2018.txt'

current\_date = 0

dates = []

# Loop through the data file and extract dates (starting from the second

# line because that is where the dates start). Append each date to the

# list of dates which will be used for the dropdown menus.

def convert\_to\_string(tup):

if tup == ' ':

tup = '/'

str = ''.join(tup)

return str

with open(filename, 'r') as f:

next(f)

for line in f:

date = (line[0:4] + " " + line[4:6] + " " + line[6:8])

if date != current\_date:

dates.append(convert\_to\_string(date))

current\_date = date

# Print out dates - note that the list is returned with strings

# instead of tuples, as is needed.

print(dates)

# Create window

self.win\_select = tk.Tk()

self.win\_select.title("Weather Data Analysis | Select Date(s)")

self.win\_select.minsize(width=500, height=250)

# Configure columns

self.win\_select.columnconfigure(0, minsize=150)

self.win\_select.columnconfigure(1, minsize=150)

self.win\_select.columnconfigure(2, minsize=150)

# Configure rows

self.win\_select.rowconfigure(0, minsize=125)

self.win\_select.rowconfigure(1, minsize=125)

# Create the StringVar and store a default value for the dropdown

# windows.

self.var = StringVar(self.win\_select)

self.var.set = dates[0]

# Create the labels.

self.win\_select.lbl\_main = tk.Label(self.win\_select,

text="Select a range of dates:",

font=('Arial', 12))

self.win\_select.lbl\_main.grid(row=0, column=1, columnspan=3)

self.dropdown\_1 = OptionMenu(self.win\_select,

self.var,

\*dates)

self.dropdown\_1.grid(row=1, column=0)

self.win\_select.lbl\_to = tk.Label(self.win\_select,

text="to",

font=('Arial', 12))

self.win\_select.lbl\_to.grid(row=1, column=1)

self.dropdown\_2 = OptionMenu(self.win\_select,

self.var,

\*dates)

self.dropdown\_2.grid(row=1, column=2)