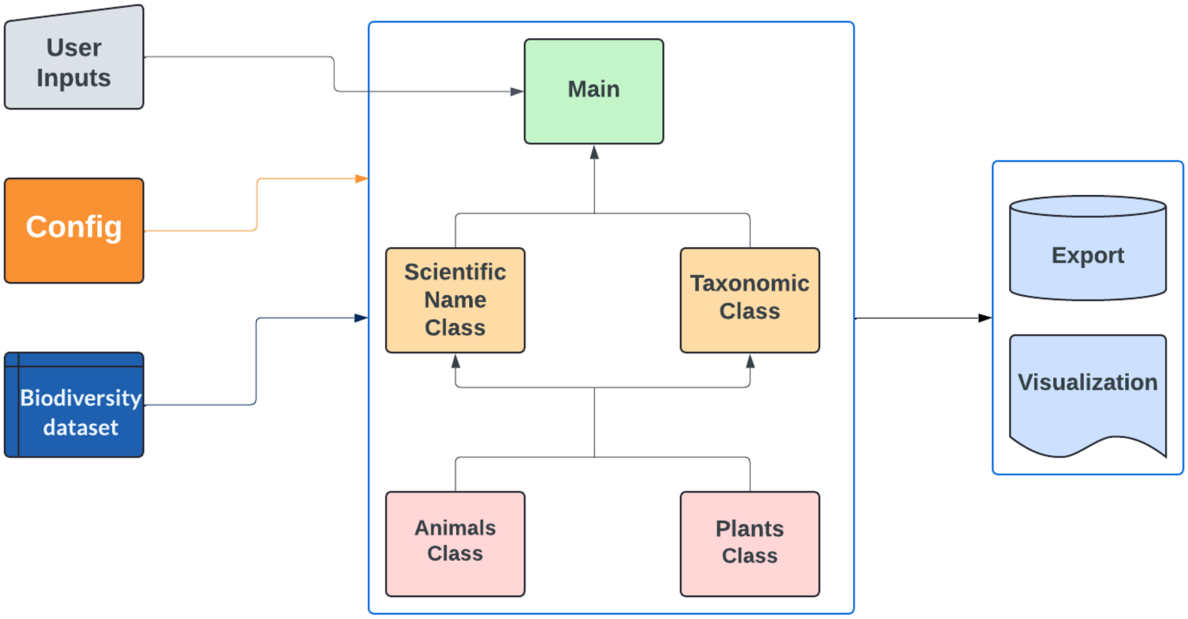
Module Communication Flow Diagram



Data format: CSV

https://catalog.data.gov/dataset/biodiversity-by-county-distribution-of-animals-plants-and-natural-communities

GitHub URL:

https://github.com/CS340-S-24-LionCoders/CS340_S_24_LionCoders

Task Progress Report

Date	Task Name	Task Description	Status	Person
04/08/2024	Flow Diagram	Create Module Communication Flow Diagram	Done	Chloe, Aubrey, Carrington, Julie
04/08/2024	Module Outline	Define the outline for each module	Done	Chloe, Aubrey, Carrington, Julie
04/06/2024	Input Data	Define Input data format	Done	Aubrey
04/06/2024	Generate/obtain dataset	Obtain or generate test input data	Done	Aubrey

Draft code

Filename: main.py

Python

```python
#This section is where we will prompt a user to select different aspects of our class which will then prompt our export file

#An example of this potential user interaction can be:

print("Welcome to our Biodiversity By Country -Ditsribution of Animals, Plants, and Natural Communities System! Please select what action you will like to take next!")

print("1) create a new Animal
       2) create a new Plant
       3)export a particular county of plants or animals
       4)export a specific taxonimc dataset
       5) export a specific scientific name class
       6) leave the system
")

inTheSystem = true

while inTheSystem:
       userTask = input("Please choice your task: ")
       if userTask == 1:
              do task 1
       else if userTask == 2:
              do task 2
       else if userTask == 3:
              do task 3
       else if userTask == 4:
              do task 4
       else if userTask == 5:
              do task 5
       else if userTask == 6:
              do task 6
       else:
              return error
```

Filename: Config.py

Python

```python
#Note: A Config file is commonly used to store parameters and settings for an application

#imports
import pandas as pd
import matplotlib.pyplot as plt
import seaborn
import numphy as np

#dataset

df = pd.read_csv('Biodiversity_by_County_-_Distribution_of_Animals_Plants_and_Natural_Communities.csv', index_col='county')



#as we continue to build, this file will get larger
```

Filename: Biodiversity_by_County_-_Distribution_of_Animals__Plants_and_Natural_Communities.csv

Python

```python
# Note: The dataset is a large file but here is a sample of what it looks like
County,Category,Taxonomic Group,Taxonomic Subgroup,Scientific Name,Common Name,Year Last Documented,NY Listing Status,Federal Listing Status,State Conservation Rank,Global Conservation Rank,Distribution Status
Albany,Animal,Amphibians,Frogs and Toads,Anaxyrus americanus,American Toad,1990-1999,Game with open season,not listed,S5,G5,Recently Confirmed
# it continues from here
```

Filename: plantsClass.py

Python
```
# The following is a rough idea of what the code should look like for
the plant class

class plantClass:
      ## must store our config restrictions in a dictionary
      config = dict()

      with open("\\Config.py") as file:
               ## looking through the config setting and place the
                  infomation in our global config
      #

      def __init(self):
            ## loading in our data into dataframe
            info =
            pd.read_csv("\\Biodiversity_by_County_-_Distribution_of_Anim
            als__Plants_and_Natural_Communities.csv")
            data = pd.DataFrame(info)

      #

      def histogramPlot():
            ## visual data in histogram plot
            plt.figure(figsize=(15,5),dpi=100)
            alpha_bar_chart = 0.75

            histogram = plt.subplot2grid((??,??),(??,??))
                                          # np.arrange(start,stop,steps
            inbetween)
            plt.hist(pasUpto19.Pclass, bins=np.arange(??,??,??),
            color='#011f4b')
            graph1.set_xticks([ #TBD # ])
            #labelling our axis and graph for PCLASS
            plt.xlabel("")
            plt.ylabel("")
            plt.title("")
      #

      def linePlot():
            ## visual data in line plot
            x =
            y =
            plt.plot(x,y)
            plt.show()
      #
#
```

Filename: animalsClass.py

Python

```python
# The following is a rough idea of what the code should look like for the animal class

class animalClass:
      ## must store our config restrictions in a dictionary
      config = dict()

      with open("\\Config.py") as file:
               ## looking through the config setting and place the
                  infomation in our global config
      #

      def __init(self):
            ## loading in our data into dataframe
            info = pd.read_csv("\\Biodiversity_by_County_-_Distribution_of_Animals__Plants_and_Natural_Communities.csv")
            data = pd.DataFrame(info)

      #

      def histogramPlot():
            ## visual data in histogram plot
            plt.figure(figsize=(15,5),dpi=100)
            alpha_bar_chart = 0.75

            histogram = plt.subplot2grid((??,??),(??,??))
```

```
                                          # np.arrange(start,stop,steps
            inbetween)
            plt.hist(pasUpto19.Pclass, bins=np.arange(??,??,??),
            color='#011f4b')
            graph1.set_xticks([ #TBD # ])
            #labelling our axis and graph for PCLASS
            plt.xlabel("")
            plt.ylabel("")
            plt.title("")
      #

      def linePlot():
            ## visual data in line plot
            x =
            y =
            plt.plot(x,y)
            plt.show()
      #
#
```

Filename: scientificNameClass.py

Python

```
# The following is a rough idea of what the code should look like for
the plant class

class scientificNameClass(animalClass,plantClass):
      ## must store our config restrictions in a dictionary
      config = dict()

      with open("\\Config.py") as file:
               ## looking through the config setting and place the
                  infomation in our global config
      #

      def __init(self):
            ## loading in our data into dataframe
```

```
        info =
        pd.read_csv("\\Biodiversity_by_County_-_Distribution_of_Anim
        als__Plants_and_Natural_Communities.csv")
        data = pd.DataFrame(info)

    #

#visuial display section
    def violinPlot():
        seaborn.set(style='whitegrid')
        dataset = seaborn.load_dataset(data)
        seaborn.violinplot(x="an x-axis value" , y = "an y-axis
        value" data=dataset)
    #

    def whiskerBoxPlot():
        ## visual data in whisker-box plot

        plt.boxplot(dataset)
        plt.show()
    #

    def scatterPlot():
        ## visual data in scatter plot
        x =
        y =
        plt.scatter(x,y)
        plt.show()
    #

#calculating section
    def calculateJointCounts():
        ##will calculate joint counts and return result
    #
    def calculateJointCounts():
        ##will calculate joint counts and return result
    #
    def calculateJointProbabilities():
        ##will calculate joint probabilities and return result
    #
    def calculateConditionalProbabilities():
        ##will calculate conditional probabilities and return result
    #
```

```
        def calculateMean():
                ##will calculate mean and return result
        #
        def calculateMedian():
                ##will calculate median and return result
        #
        def calculateSTD():
                ##will calculate STD and return result
        #

#categorial attribute section

        def obtainSpecificValue(String "askedValue"):
                ##will return the asked value
        #
        def generatePermutationsOfNames():
                ##will return an ordered arrangement of names
        #
        def generateCombinationsOfNames():
                ##will return a unordered arrangement of names
        #
```

Filename: taxonomicClass.py

Python

```python
class taxonomicClass(animalClass,plantClass):
      ## must store our config restrictions in a dictionary
      config = dict()

      with open("\\Config.py") as file:
               ## looking through the config setting and place the
                  infomation in our global config
      #

      def __init(self):
            ## loading in our data into dataframe
            info =
            pd.read_csv("\\Biodiversity_by_County_-_Distribution_of_Animals__Plants_and_Natural_Communities.csv")
            data = pd.DataFrame(info)
```

```
    #

#visuial display section
    def violinPlot():
        seaborn.set(style='whitegrid')
        dataset = seaborn.load_dataset(data)
        seaborn.violinplot(x="an x-axis value" , y = "an y-axis
        value" data=dataset)
    #

    def whiskerBoxPlot():
        ## visual data in whisker-box plot

        plt.boxplot(dataset)
        plt.show()
    #

    def scatterPlot():
        ## visual data in scatter plot
        x =
        y =
        plt.scatter(x,y)
        plt.show()
    #

#calculating section
    def calculateJointCounts():
        ##will calculate joint counts and return result
    #
    def calculateJointCounts():
        ##will calculate joint counts and return result
    #
    def calculateJointProbabilities():
        ##will calculate joint probabilities and return result
    #
    def calculateConditionalProbabilities():
        ##will calculate conditional probabilities and return result
    #
    def calculateMean():
        ##will calculate mean and return result
    #
    def calculateMedian():
```

```
                ##will calculate median and return result
        #
        def calculateSTD():
                ##will calculate STD and return result
        #

#categorial attribute section

        def obtainSpecificValue(String "askedValue"):
                ##will return the asked value
        #
        def generatePermutationsOfNames():
                ##will return an ordered arrangement of names
        #
        def generateCombinationsOfNames():
                ##will return a unordered arrangement of names
        #
```