Main.py

```
import io

import gladysUserInterface as userInterface

"""
   This is the main code.  This is the .py file that you run to execute and
   run/start the application.
   Students do NOT need to change this code.
"""

userInterface.start()

# this is an example of the help command on a module.  You can get help on
# any python module The docstring documentation is what is displayed.
# help(userInterface)
```

gladysUserLogin.py

```
import io
"""
 Student Name: Tuari Vo
 Module: gladysUserLogin
 Description: This module logs the users in through the user login and password
"""

def login():
   userIn = False

   while (not userIn):
     print("Hello Please Enter Your Info")
     userInput1 = input("Enter Login: ")
     userInput2 = input("Enter Password: ")

     if userInput1 == 'blt@gwm.com':
       userIn = True

     if userInput2 == '123':
       userIn = True

     if userIn == True:
       print("\n")
```

```
        print("Successfully Login!")
        print("\n")

    else:
        print("Error: " + userInput1 + " is incorrect")
        print("Error: " + userInput2 + " is incorrect")
```

gladysUserInterface.py

```
"""
   Student: Lavon Nguyen,Tuari Vo
   Module: gladysInterface
   Description: This module presents a text-based menu for users thats calls the other
modules together.
"""

import io

import gladysCompute as compute
import gladysSatellite as satellite
import gladysUserLogin as userLogin

"""
   prints the menu
"""
def runTests():
   """Tests module function
   """

   print("Running a few tests")

   average = compute.gpsAverage (1,2)
   print("average = ", average)

   print("Test Completed!")

def start():
   """
       Logs the user in and runs the app
   """
   userName = userLogin.login()
   runApp(userName)
```

```
def runApp(userName):
    """
        Runs the Glady West Map App
    """
    xCurrPos = -1
    yCurrPos = -1
    xDestPos = -1
    yDestPos = -1
    dist = 0

    #Loops until user types q
    userQuit = False

    while(not userQuit):
        #menu interface

        print('\n----------------------')
        print('Gladys West Map App')
        print('--------------------------')
        print('user = ', userName, '\n', sep='')
        print('curent position      : x=', xCurrPos,  ', y=', yCurrPos)
        print('destination position : x=', xDestPos,  ', y=', yDestPos)
        print('distance             :', round(dist, 2), '\n')

        print("[c] to set current position")
        print("[d] to set destination position")
        print("[m] to map- to tell the distance")
        print("[t] to run module test")
        print("[q] to quit")

        # gets the first character letter of input
        userInput = input("\nEnter a command:")
        lowerInput = userInput.lower()
        firstChar = lowerInput [0:1]

        if firstChar == 'c':
            # sets the current position
            xInput = input('Enter a x position:')
            yInput = input('Enter a y position:')
            xCurrPos = int(xInput)
            yCurrPos = int(yInput)
```

```
            current = [xCurrPos, yCurrPos]
            destination = [xDestPos, yDestPos]
            dist = compute.distance(current, destination)
        elif firstChar == 'd':
            # sets the destination position
            xInput = input('Enter a x position:')
            yInput = input('Enter a y position:')
            xDestPos = int(xInput)
            yDestPos = int(yInput)
            current = [xCurrPos, yCurrPos]
            destination = [xDestPos, yDestPos]
            dist = compute.distance(current,destination)
        elif firstChar == 'm':
            # Map, which tells the distance
            print('\n-------------------')
            print('distance = ', dist)
            print('-------------------\n')
        elif firstChar == 'q':
            # Quits the App
            userQuit = True
        elif firstChar == 't':
            # Runs some tests
            runTests()

        else:
            print("ERROR: " + firstChar + "is not a valid command")

        print("\n")
        print("Success!")
        print("Thanks for using the Glady West Map App!")
        print("\n")
```

gladysSatellite.py

```
import io
import json

"""
   Student: Lavon Nguyen, Tuari Vo
   Module: gladysSatellite
```

```
    Description: This module takes the files and reads the data in it, and calculates
the x and y inputs
"""


def readSat(sat, pathToJSONDataFiles):
    """
        reads satellite data from a json file
        Students do NOT need to change the readSat function.
    """

    # data file path
    fileName = sat + "-satellite.json"
    filePath = pathToJSONDataFiles + "/" + fileName

    # open the file
    try:
        fileHandle = open(filePath)
    except IOError:
        print("ERROR: Unable to open the file " + filePath)
        raise IOError

    # print("filePath = ", filePath)

    # read the file
    data = json.load(fileHandle)

    return data


def gpsValue(x, y, sat):
    """
    reads the json file and returns the value that is defined for it
    """
    pathToJSONDataFiles = '/Users/azntv/Documents/GitHub/Glady-West-Map-Project/'


    # read the satellite data
    data = readSat(sat, pathToJSONDataFiles)

    value = 0
    for item in data:
```

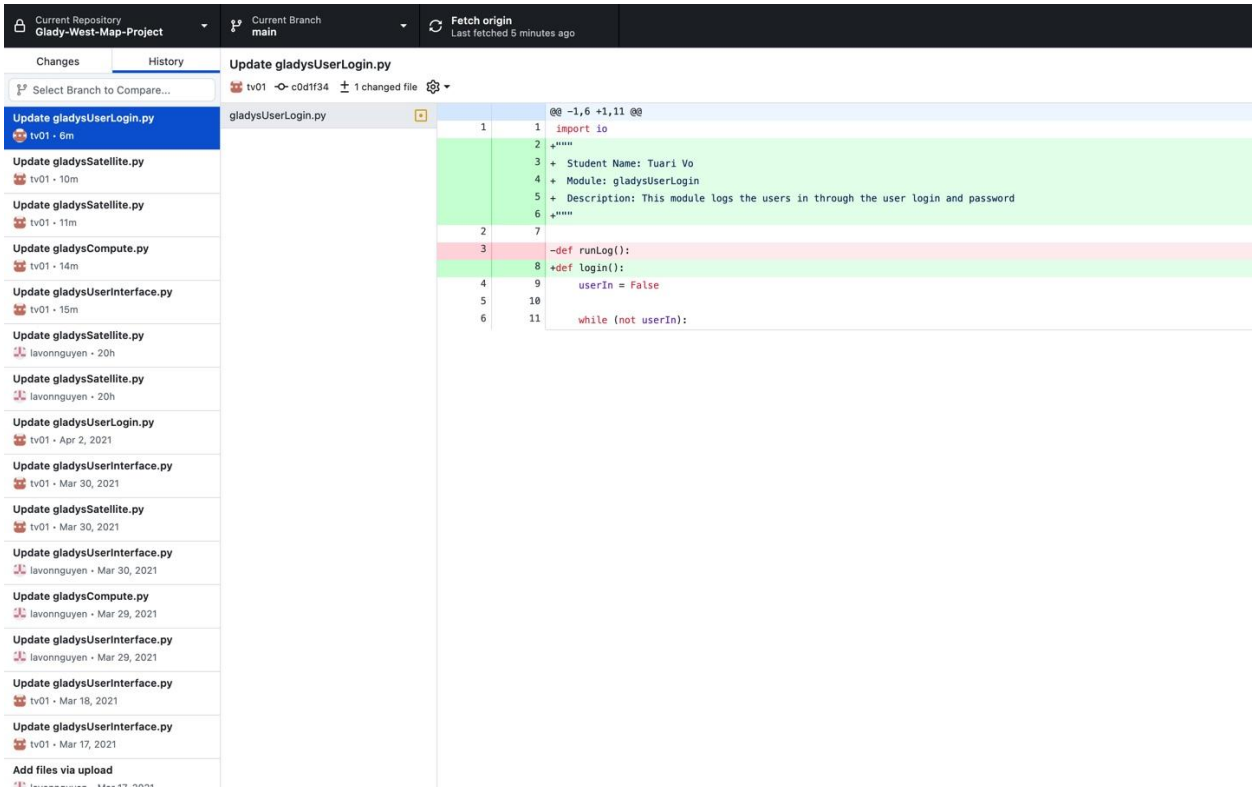
```
            if item['x'] == x and item['y'] == y:
                value = item['value']

    if value == 0:
        print('ERROR: No value for x =', x, 'and', 'y =',
                y, 'in satellite =', sat)

    return value
```

gladysCompute.py

```
import io

import gladysSatellite as satellite



"""
    Student: Brittany Quach
    Module: gladysCompute
    Description: This module does calculates the formula for gpsAverage and distance
"""

def gpsAverage (x,y):
    """
        Calculates the gpsAverage formula
    """
    latitude = satellite.gpsValue(x,y, 'latitude')
    longitude = satellite.gpsValue(x,y, 'longitude')
    altitude = satellite.gpsValue(x,y, 'altitude')
    time = satellite.gpsValue(x,y, 'time')

    gpsValuesSum = latitude + longitude + altitude + time
    average = gpsValuesSum / 4

    return average

def distance (current, destination):
    """
        Calculates the distance formula
    """
    currGpsAverage = gpsAverage(current[0], current[1])
    destGpsAverage = gpsAverage(destination[0], destination[1])
```

```
    dist = (currGpsAverage ** 2 + destGpsAverage ** 2) ** 0.5

    return dist
```

GitHub History and Commits:



Demo Video of Software Running:

https://www.youtube.com/watch?v=YCvr4ZInKRM