COIT20245: INTRODUCTION TO PROGRAMMING (HT1, 2024)

**ASSIGNMENT 2 – PROJECT**

**Queensland Wildlife Sightings**

**Present to:** (Tutor)

**Due date:** 19 May 2024

**LEADER**

**Name:** Bittu Patel

**Student ID:** 12262831

**MEMBER 1**

**Name:** Tejas Patel

**Student ID:**12260683

Table of Content

**OVERVIEW AND PURPOSE**

The Queensland Wildlife Sightings Python web application aims to streamline the observation, documentation, and analysis of wildlife sightings throughout Queensland, Australia. Utilizing contemporary web technologies, this application offers an easy-to-use platform where users can report sightings, access real-time wildlife data, and connect with the local ecological community.

**Tools and technology used to create this application.**

**Backend Technologies:**Python

**Mentioned API:** Services that provide the map data and geolocation capabilities.

**Development and Collaboration Tools:**

**Git:** Version control system for tracking changes in the source code.

**GitHub:** software to host a code repository and facilitate cooperation.

**GitHub Link:**

[**https://github.com/patel2921/COIT20245-Assignment-2.git**](https://github.com/patel2921/COIT20245-Assignment-2.git)**/**

**Google Drive Link:**[**https://drive.google.com/drive/folders/1IyilkEhjckJD99gEySFu7zD1iqldrwAK?usp=sharing**](https://drive.google.com/drive/folders/1IyilkEhjckJD99gEySFu7zD1iqldrwAK?usp=sharing)**.**

**Testcases and Output**

**Task 1 Display Menu**

* **Code (SS)A screenshot of a computer program

  Description automatically generated**
* **Input/Output (SS)**

**A screen shot of a computer

Description automatically generated**

* **Limitation and Bug: -** In this task there is not any bug and error occur**.**

**Task 2 User Input**

* **Code**

**A screen shot of a computer program

Description automatically generated**

* **Test Input/Output**

**A screen shot of a computer

Description automatically generated**

* **Limitation and bugs: -** In this Task 2 there is not any bug and error occur. This task limited to only display menu help and exit.

**Task 3 List Species in City**

* A screen shot of a computer program

  Description automatically generated**Code**
* **Test input and Output**

1. **Input :-** user input the city name, Here you can show the Brisbane data.

**A screenshot of a computer

Description automatically generated**

* **Limitation and Bugs: -**Here you can show clearly when the user input any city for find species. so sometime API does not get accurate data. Limitation of this function is it fetch only city data.

**Here is the solution of the API: -**

https://us1.locationiq.com/v1/search?key=pk.37e94aa3017ca984d06aff04b69037ef&q={city}&format=json&

**A computer screen shot of text

Description automatically generated**

**Task 4** **List Animal** **Sightings in City**

* **A screen shot of a computer

  Description automatically generatedCode**
* **Test Input/ Output**
* User input City name for finding List Animal Sightings in City. Here the city name is Brisbane.

**A screenshot of a computer

Description automatically generated**

* **Limitation and Bugs: -** Here you can show I found same error similar like previous one. When the user input any city for find species. so sometime API does not get accurate data. Limitation of this function is it fetch only Sighting’s city data.

**A computer screen shot of text

Description automatically generated**

**Task 5** **List Venomous Animal Sightings in a City**

* **A screen shot of a computer program

  Description automatically generatedCode**
* **Test input / Output**
  + User input City name for finding Animal List Venomous Animal Sightings in a City. Here the city name is Queensland.

**A screenshot of a computer

Description automatically generated**

* **Limitation and Bugs: -**Here you can show I found same error similar like previous one. When the user input any city for find species. so sometime API does not get accurate data. Limitation of this function is it fetch only Sighting’s city data.

1. **Error Handling:** The code does not handle cases where user input may not be in the expected format, or when the city or species does not exist in the data.
2. **Efficiency**: The code could be optimized for better performance, especially when dealing with large datasets

**A screen shot of a computer program

Description automatically generated**

**Task 6 Add a GSP Stub**

* **Code**

**A screen shot of a computer program

Description automatically generated**

* A computer screen with colorful text

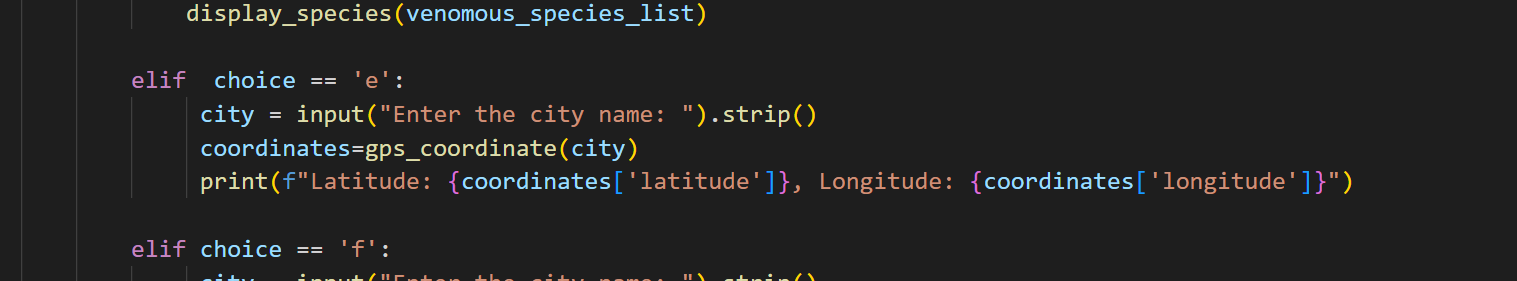
  Description automatically generated**Test input/Output: -** In this task we make the gps\_coordinate(city)function that return the longitude and latitude of city that enter by the user. Here you can show the output of the different city’s latitude and longitude.
* **Limitation and Bugs: -**In this task there is no bugs found. Here function only output 3 or 4 cities latitude and longitude, because of in the API there are only 3 or 4 cities data**.**
  1. **Hardcoded API Key:** The API key is hardcoded, which is not secure for production use.
  2. **API Rate Limiting:** The function does not handle rate limiting errors from the API.
  3. **Unit Testing:** The current test cases are commented out, and more extensive testing with a variety of cities and error scenarios would be beneficial.

**Task 7 GPS Webservice Module**

* **Code**

**A screen shot of a computer program

Description automatically generated**

****

* **Test input/Output:** - When user input the city name it returns the city latitude and longitude. Here user input city name is Queensland, so latitude is: -16.92 and longitude is 145.777**.**

**A screenshot of a computer program

Description automatically generated**

* **Limitation and Bugs: -** When the user input other city name so it will throw the error in code because in API has limited city latitude and longitude.

1. **Hardcoded API Key:** The API key is hardcoded, which is not secure or flexible for different environments.
2. **Response Validation:** The function does not adequately validate the response data, which could lead to exceptions if the structure of the response changes or if the API returns unexpected data.

**A screenshot of a computer program

Description automatically generated**

**Task 8** **Wildlife Module Get Species List**

* **A screen shot of a computer program

  Description automatically generatedCode**
* **Test Input/Output: -**User input the latitude, longitude, and radius so it shows the wildlife module species list.

A screenshot of a computer

Description automatically generated

* **Limitation and Bugs: -** If user input wrong data like do not match with city coordinates and tax\_on\_id. so, it shows errors.

1. **Handling of JSON Response:**

If the API response does not contain the expected keys or structure, the code will raise a “KeyError”. There is no error handling for unexpected API responses or network errors.

**A screen shot of a computer code

Description automatically generated**

**Task 9 Wildlife Module Get** **Surveys by Species**

* **Code**

**A computer screen shot of text

Description automatically generated**

**A computer screen shot of a program code

Description automatically generated**

* **Test input/output: -**User input the city coordinates and tax\_on\_id if is match with API data so. It is showing the list of surveys by species.

**A screenshot of a computer

Description automatically generated**

* **Limitation and Bugs: -** This code limitation is if user input the outside of the data, it cannot be print anything. It is showing the only empty list**.**

**A screen shot of a computer

Description automatically generated**

**Task 10 Sort**

* **Code**

**A screen shot of a computer program

Description automatically generated**

**A computer screen shot of a program code

Description automatically generated**

* **Test input/output: -** User should input the city name and Tax\_on\_id. Then it is showing the list of common data with sorting by date (minimum start date).
* **Sort by date:** -
* TaxonID: 767
* Start Date: **1770-05-23**
* Locality Details: Cairns
* TaxonID: 767
* Start Date: **1976-06-09**
* Locality Details: Freshwater Creek, 1.2 km W of Mooroobool Peak
* TaxonID: 767
* Start Date: **2017-09-06**
* Locality Details: Cairns

Here you can show the sort by date all species

A computer screen shot of a black screen

Description automatically generated

* **Limitation and Bugs: -**This code limitation is if user input the outside of the data, it cannot be print anything. It is showing the only empty list.

**A screen shot of a computer

Description automatically generated**