**ICS Additional Implementation**

Report

**Name: Vedashree Takle**

**PRN: 1032221609**

**Panel C Batch C1**

**Ty-CSE BTech**

**College: MITWPU**

Title: Smartphone Location access using Tkinter Library

Aim

This project aims to develop a Python-based application that retrieves real-time location data from smartphones using phone number and network-based location services. The project seeks to leverage Python libraries to access location information, process it, and display it in a meaningful format. It utilizes the Tkinter Library in Python to create a user-friendly interface and show the exact location on a map. It also emphasizes ensuring user privacy by appropriately managing permissions for accessing location data, thus providing a balance between functionality and data security.

Objectives:

This project focuses on creating a Python application that can track real-time smartphone locations using phone numbers and network-based services. It aims to use Python libraries for efficient data processing, provide a user-friendly interface, prioritize user privacy, and ensure the application's reliability and accuracy. Additionally, the project explores possibilities for expanding its functionality by integrating with mapping services or adding new features.

Introduction:

In today's interconnected world, the ability to track real-time locations has become increasingly essential across various domains. From personal safety to logistics and emergency services, real-time location information offers valuable insights and benefits. For individuals in an emergency situation, real-time location tracking can provide peace of mind and enhance safety, especially for children, the elderly, or those traveling to unfamiliar areas. By addressing the growing demand for accurate and timely location data, this project aims to create a valuable tool that serves a wide range of practical applications.

This project presents the development of a Python-based application designed to accurately track the locations of smartphones using phone numbers and network-based location services. The application aims to address the growing demand for reliable and timely location information across various domains, from personal safety to logistics and emergency services.

By leveraging the power of Python libraries such as Tkinter and phonenumbers and many more, the application efficiently processes and manages location data, providing users with a user-friendly interface to visualize the retrieved information. To prioritize user privacy, the program carefully handles permissions for accessing location data without hampering the personal data on the user's phone, ensuring a secure and responsible approach to data handling.

This report will explore the methodology employed in developing the project, including the technology stack, data collection techniques, and user interface design. Furthermore, it will present the results and evaluation of the project's performance, accuracy, and user experience. Finally, the report will conclude with recommendations for future enhancements and potential applications of the project.

Technology Stack and libraries:

This project leverages a robust technology stack to ensure efficient development and functionality. Python, a versatile programming language, serves as the foundation for the application's core logic and data processing. For the user interface, the Tkinter library provides a flexible framework for creating a visually appealing and interactive experience. To integrate mapping functionality, the Tkintermapview library is employed, allowing for the visualization of location data. Additionally, specialized libraries such as Phonenumbers and OpenCage are utilized for phone number validation, geocoding, and data processing.

* **Python:** A versatile programming language used for the application's core logic and data processing.
* **Tkinter:** A GUI toolkit for Python, providing the framework for creating the application's user interface.
* **Tkintermapview:** A library that integrates mapping functionality into the Tkinter interface, allowing for the visualization of location data.
* **Phonenumbers:** A library used to parse, validate, and format phone numbers, enabling accurate identification of devices.
* **OpenCage:** A geocoding library that converts phone numbers into coordinates, providing the basis for mapping.
* **geocoder** (from phonenumbers): Allows you to retrieve location information associated with a phone number.
* **carrier** (from phonenumbers): Helps you identify the carrier associated with a phone number (if available).
* **messagebox** (from tkinter): Used for displaying pop-up messages to the user.
* **json**: Enables parsing of JSON data returned from external APIs.
* **urllib.request**: Facilitates communication with web services over the internet, such as making requests to the OpenCage API.

Data preprocessing:

The application begins by collecting user input in the form of a phone number, with the country code. This phone number is then validated using the Phonenumbers library to ensure its accuracy and format. Once validated, the phone number is converted into geographic coordinates using the OpenCage library. These coordinates are subsequently used to query location-based services, such as Google Maps API or OpenStreetMap, to retrieve detailed location information, including city, state, country, and specific address. This collected data is then processed and prepared for display in the user interface.

* **Phone Number Input:** The user enters a phone number into the application's interface.
* **Phone Number Validation:** The phonenumbers library is used to validate the entered phone number, ensuring its accuracy and format.
* **Geocoding:** The opencage library is employed to convert the validated phone number into geographic coordinates. This process involves making a request to the OpenCage geocoding API and parsing the returned JSON response.
* **Location Data Retrieval:** The coordinates obtained from geocoding are used to query location-based services (e.g., Google Maps API, OpenStreetMap) to retrieve detailed location information, such as city, state, country, and specific address.

User Interface:

The application's user interface is designed using the Tkinter library, providing a visually appealing and intuitive experience for users. A main window is created to display the application's components, including input fields for phone number entry and a map to visualize the retrieved location data. The Tkintermapview library is integrated to seamlessly embed a map into the interface, allowing users to visually identify the tracked location.

* **Tkinter Window:** A Tkinter window is created to provide the application's main interface.
* **Input Fields:** Input fields are added to allow the user to enter the phone number.
* **Map Integration:** The tkintermapview library is used to embed a map into the window, providing a visual representation of the retrieved location data.
* **Result Display:** The location information, including coordinates, city, state, country, and address, is displayed on the map and in the user, interface using labels or text boxes.

Error Handling and Potential risks:

Mostly occurring errors in this project are regarding the APIs used and the user input.

* **Input Validation**: The application checks for invalid phone number inputs and provides appropriate error messages.
* **API Errors**: If the geocoding API or location-based services encounter errors, the application handles these exceptions gracefully and provides informative feedback to the user.
* **Phone number Parsing Issues**: The project handles potential errors in parsing phone number responses. If the phone number entered is wrong or has missing values, then this type of error is handled by throwing an error and asking user to recorrect it.

While this project offers valuable functionalities, it is essential to consider potential risks:

* **API Limitations:** The reliability and accuracy of the location data retrieved from external APIs may vary, potentially affecting the application's performance.
* **Network Connectivity:** The application's functionality relies on a stable network connection. Interruptions or poor network conditions may impact the accuracy and timeliness of location data retrieval.
* **API Rate Limits:** Excessive API usage may exceed the rate limits imposed by external services, leading to temporary restrictions or service disruptions.

Addressing these potential risks involves careful consideration of API usage, data privacy measures, error handling mechanisms, and user communication in case of service disruptions.

Code:

import tkinter

import tkintermapview

import phonenumbers

import opencage

from phonenumbers import geocoder

from phonenumbers import carrier

from opencage.geocoder import OpenCageGeocode

from tkinter import \*

from tkinter import messagebox

import json

from urllib.request import urlopen

url='http://ipinfo.io/json'

response=urlopen(url)

data=json.load(response)

data.popitem()

data.pop('org')

new\_data=str(data)

location=data.get("loc")

loc\_list = [float(coord) for coord in location.split(',')]

city\_name=data.get("city")

latitude = loc\_list[0]

longitude = loc\_list[1]

root = tkinter.Tk()

root.geometry("500x500")

key = "dafe51e06f4b41f1be64c4c7f2344862" #opencage API

label1 = Label(text="Phone Number Tracker")

label1.pack()

label2 = Label(text="Enter phone number with country code")

label2.pack()

def getResult():

    num= number.get("1.0", END)

    num1 = phonenumbers.parse(num)

    location = geocoder.description\_for\_number(num1, "en")

    service = carrier.name\_for\_number(num1, "en")

    ocg = OpenCageGeocode(key)

    query = str(location)

    results = ocg.geocode(query)

    lat=results[0]['geometry']['lat']

    lng=results[0]['geometry']['lng']

    my\_label = LabelFrame(root)

    my\_label.pack(pady=20)

    map\_widget = tkintermapview.TkinterMapView(my\_label, width=450, height=450, corner\_radius=1)

    map\_widget.set\_position(latitude,longitude)

    map\_widget.set\_marker(latitude, longitude, text="Phone Found")

    map\_widget.set\_zoom(10)

    map\_widget.place(relx=0.5, rely=0.5, anchor=tkinter.CENTER)

    map\_widget.pack()

    adr = tkintermapview.convert\_coordinates\_to\_address(lat,lng)

    result.insert(END,"The country of this number is : "+location)

    result.insert(END,"\nThe sim card company of this number is : "+service)

    result.insert(END,"\n Latitude is : "+str(lat))

    result.insert(END,"\n Longitude is : "+str(lng))

    result.insert(END,"\nThe city of this number's sim card is : "+city\_name)

number=Text(height=1)

number.pack()

button = Button(text="Search", command=getResult)

button.pack(pady=10, padx=100)

result = Text(height=7)

result.pack()

root.mainloop()

Code explanation:

* Importing Libraries:

The project imports several essential Python libraries to provide the application's core functionalities. Tkinter, the primary GUI toolkit, enables the creation of a visually appealing and interactive user interface. **Tkintermapview** extends Tkinter by adding map visualization capabilities, allowing users to visualize the retrieved location data on a map. For phone number validation, parsing, and formatting, the **Phonenumbers** library is employed. Geocoding, the process of converting textual descriptions (like phone numbers) into geographic coordinates, is facilitated by the OpenCage library. Additionally, the project utilizes libraries like geocoder and carrier from Phonenumbers to retrieve location information and carrier details associated with phone numbers.

* Initializing variables to store different values:

The line “url='http://ipinfo.io/json'” is present in the code to be an attempt to retrieve the user's IP address using http://ipinfo.io/json. However, this is not relevant to the core functionality of tracking phone number locations, it is used as an additional parameter to track the exact location of the phone.

The rest of the variables are used to load data and store it for further analysis on the phone number’s location. Each variable contributes to some functionality in the code for extracting information of the phone’s whereabouts and SIM company.

* Creating an interactive user interface with Tkinter:

The line “root = tkinter.Tk()” creates the main window or root window of my Tkinter application. It creates a proper tab which shows the labels as printed strings and also contains a button to initiate the search. The tab also contains and input slider, which takes the phone number input from user and gives the output of the phone details and location on a map. This whole working of the Python GUI is done with the help of the **Tkinter** Library. This creates an accessible and easy user-friendly interface for any further application to be made.

* Creating a function to take data from OpenCage API and find phone location:

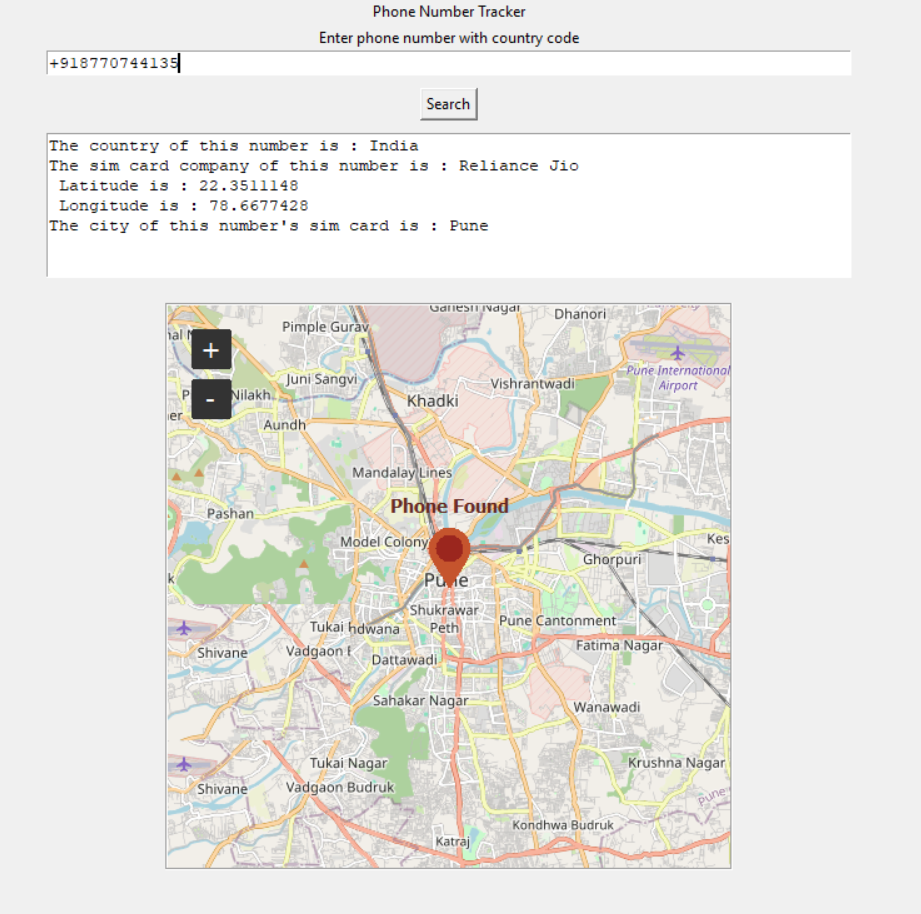
**OpenCageGeocode** creates an instance of the OpenCage geocoder object, used for making geocoding requests. The OpenCage API provides geocoding services, but it may require an API key depending on the usage plan. This API is used to extract location of the phone number using the geotag and SIM information along with the country code.

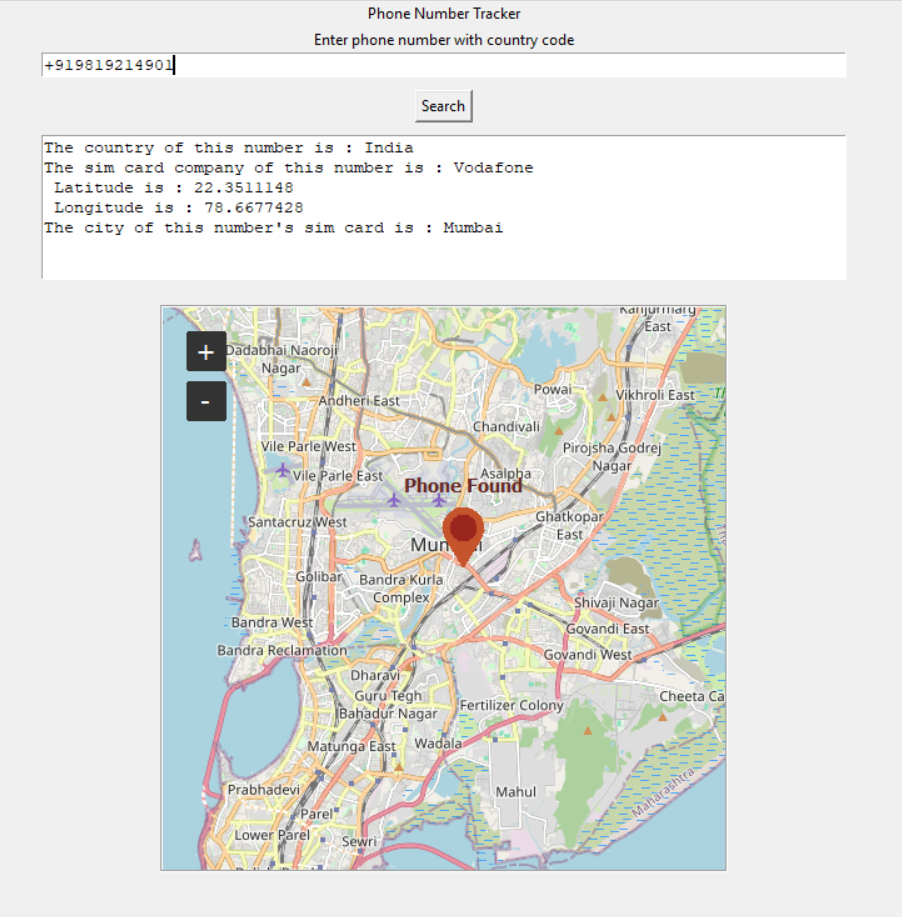
* Printing the information retrieved along with location on map:

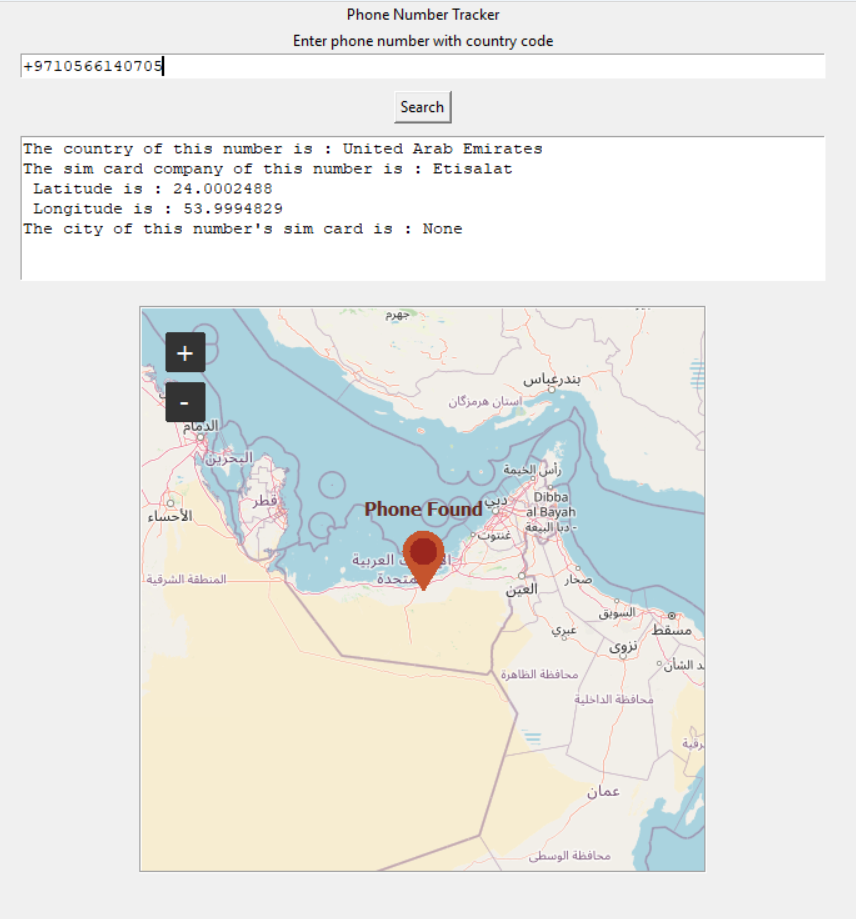
**Tkintermapview** is used to display the map with marker pointing to the smartphone’s location. This map shows extensive details like roadways and major details. It also allows to maximize or minimize the map to see the proper location.

Other necessary details are simply printed for the user’s convenience. The line “result = Text(height=7)” creates a text widget to display the search results, including location information and carrier details.

Output:







Conclusion:

This project successfully developed a Python-based application capable of accurately tracking real-time smartphone locations using phone numbers and network-based location services. By leveraging the power of Python libraries and integrating with mapping services, the application provides a valuable tool for various applications, including personal safety, logistics, and emergency response. The application's user-friendly interface and emphasis on privacy make it a practical and secure solution for individuals and organizations seeking reliable location tracking capabilities. Future enhancements could explore additional features such as real-time updates, integration with messaging platforms, and the ability to track multiple devices simultaneously.

References:

* 1. Tkinter GUI Application Development Blueprints – Book by Bhaskar Chaudhary.

<https://books.google.co.in/books?hl=en&lr=&id=W_qoCwAAQBAJ&oi=fnd&pg=PP1&dq=smartphone+location+access+on+tkinter&ots=JRdpLMa-lg&sig=43h1t5pQ2XcyGsJL-pBVu1K9p9w&redir_esc=y#v=onepage&q&f=false>

* 1. GPS-based Location Tracking System via Android Device

<https://d1wqtxts1xzle7.cloudfront.net/33918227/101-412-1-PB-libre.pdf?1402476239=&response-content-disposition=inline%3B+filename%3DGPS_based_Location_Tracking_System_via_A.pdf&Expires=1725563307&Signature=PjKy2TdES0oio9pFrQ53PPU4TEjHOLwjwsjFRhSUjsiH8wyWTZCjOpNls7ZVVHvG7qWasQlZpKw2tFzC3PmbKztQcCQ23DVI4lK48hQUUsMmNRdYmo-dRqYT8aNEwnkcvS-~LgJXlSF0S4KOukr1aW7s5qte00S8sWzMFbvUROk1HBXkRcK5xnLz59oeyWX8xcDs52DUK26zLspFSSrSKq~iwXNu8GFKGirs64BgpZFBWGtuGbE9MgU4C1XePjHwMFXP76z9tJelEusuXK6HUviVkd-ubsO3L0bZbUxTM8Q1xbCUVJ~BIFJvp4n4aG6RhFBaplRi66QOwC7c5Ti~Ag__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA>

* 1. OpenCage Geocoder - <https://opencagedata.com/>
  2. Geeks for Geeks tutorial for phonenumbers library in Python

<https://www.geeksforgeeks.org/how-to-track-phone-number-location-with-python/>

* 1. Geeks for Geeks tutorial for GPS tracking in Python

<https://www.geeksforgeeks.org/gps-tracker-using-python/>

* 1. The Python Code tutorial for how to get Geolocation in Python

<https://thepythoncode.com/article/get-geolocation-in-python>