ESCORT AMIGO

- BY TEAM 2

GROUP MEMBERS: -

1. VISHAL SRIVASTAVA (20BCE10571)
2. ANSHIKA SHARMA (20BCE10634)
3. SHUBHAM SINGH (20BCE10696)
4. NAINA SHARMA (20BCE10811)
5. SHUBH PATHAK (20BCE11023)

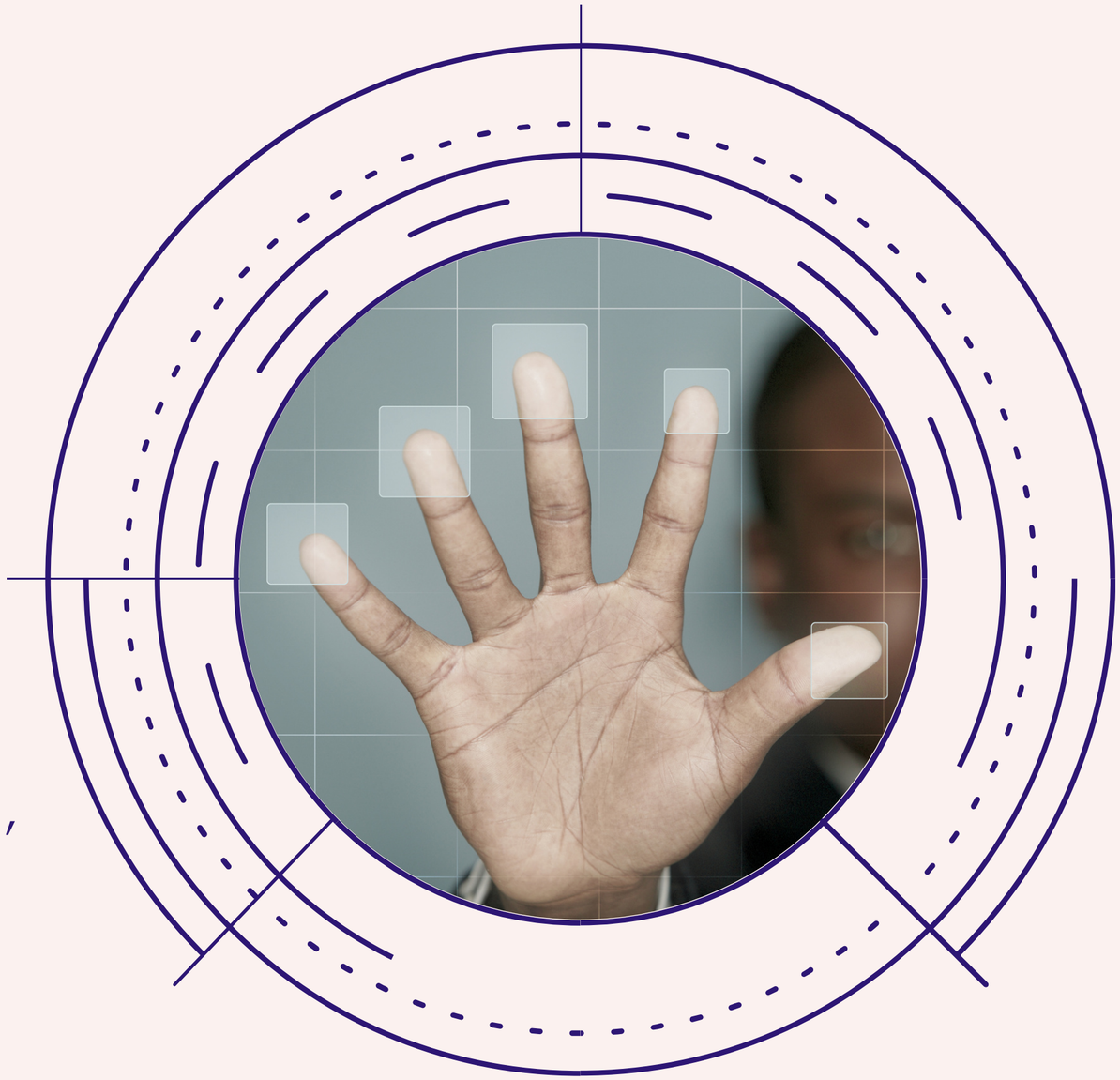
TEAM GUIDE - DR. J AMUDHAVEL

INTRODUCTION

The Escort Amigo System is a python based application. The objective of this project is to develop a system that automates the processes and activities of a travel guide or agency. In the present system a customer has to approach various agencies to find details of places. This often requires a lot of time and effort. We provide approach skills to critically examine how a tourist visits and what they wants. In this project, We will make an easier task of searching tourist places, hotels, local best foods, weather of that place, location, time zone and many more.

EXISTING WORK WITH LIMITATIONS

- In the present system a customer has to approach various agencies to find details of places. This often requires a lot of time and effort.
- A customer may not get the desired information from these offices and often the customer may be misguided.
- There are lots of redirects like When you're searching for a vacation and you get redirected to another page or website, it's annoying.



PROPOSED WORK AND METHODOLOGY

- The proposed system is a python based application and maintains a centralized repository of all related information.
- The system allows one to easily access the relevant information required to make a perfect vacation plan.
- Users can decide about places they want to visit and search best hotels, good local foods and most importantly list of tourist places.

NOVELTY OF THE PROJECT

- IT IS UNIQUE COMPARED TO OTHER WEBSITES AND SYSTEMS BECAUSE IN THIS WE COVER ALL THE TOURISTS HUB. OTHER WEBSITES ONLY SELECT FAMOUS OR LIMITED TOURIST PLACES.
- IT IS ALL IN ONE TRAVEL PLAN MAKER SYSTEM.
- THEY CAN GET ALL THEIR DESIRED INFORMATION BY SEARCHING ON OUR SYSTEM RATHER THAN GOING TO DIFFERENT AGENCIES WHERE THEY MIGHT NOT GET THEIR DESIRED INFORMATION.

Real time usage

As the customer ask for the details of a particular tour, they need to visit different websites and agencies to acquire information and there's a possibility that they get misguided or doesn't get desired information and also time consuming whereas our Escort Amigo System provides the detailed information of the related place where tourist/customer wants to go, time zone of that place, local cuisines, best hotels, tourists spots, navigation system and much more and it also consumes very less time.

Hardware & Software Requirements

Hardware system configuration: -

Any Contemporary PC

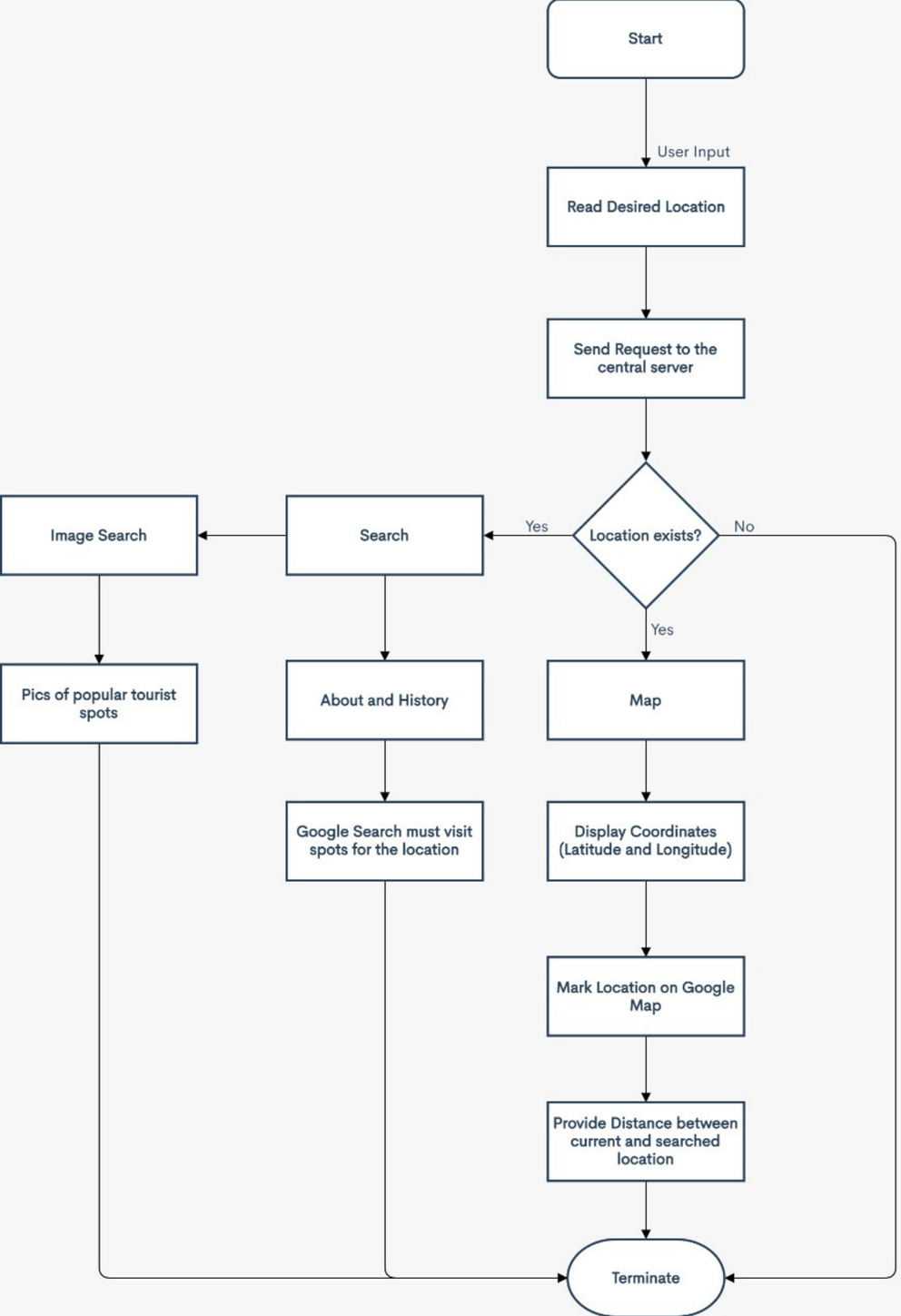
Software system configuration: -

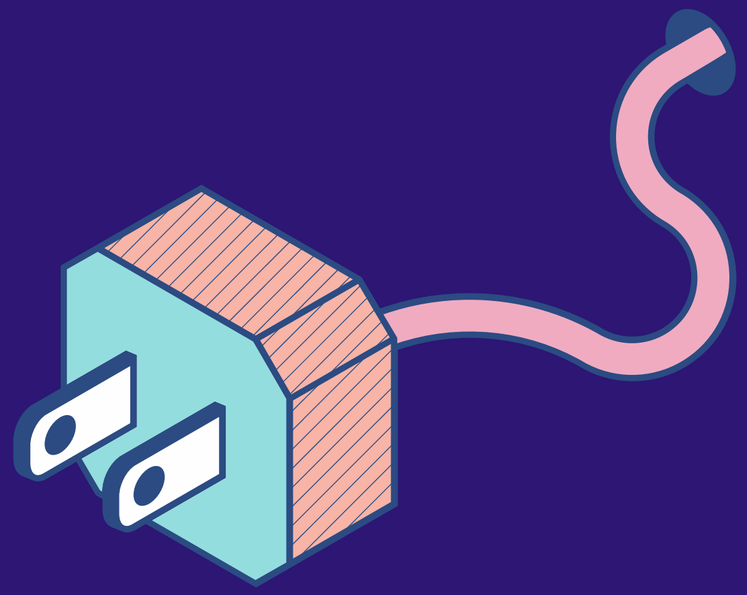
Operating system -> Windows /UNIX/LINUX/Mac-OS

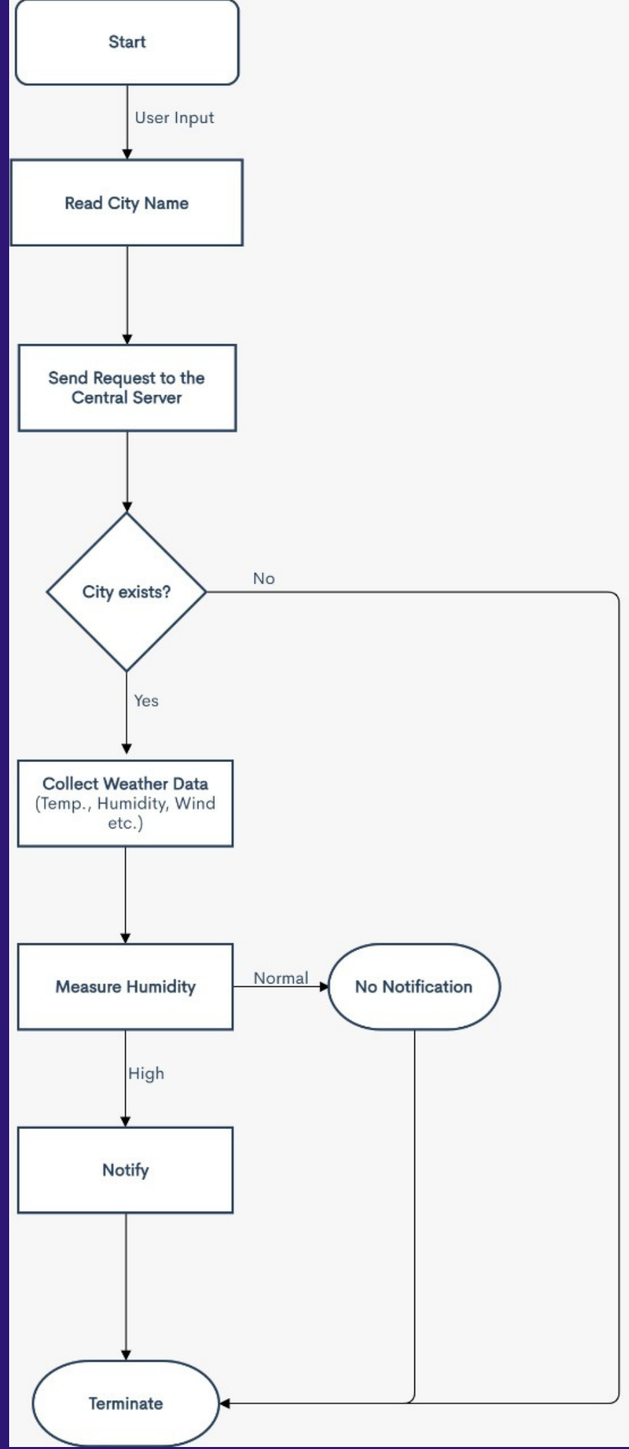
Python 3

MySQL

Overall Architecture diagram and flow diagram

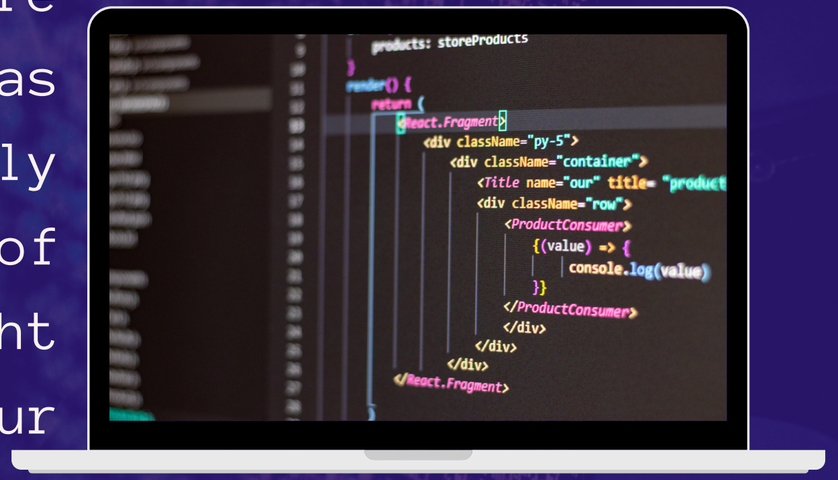






LITERATURE REVIEW

In order to get a broader picture of the field of tourism and technology, we have been reviewing literature in the fields of tourist behavior, software development, user requirement handling as well as articles and project reports describing previously developed software related to tourism. The review of these previous works in this field provided an insight into user's attitudes towards different features in our software since these project reports and articles described not only the systems but also evaluations and user trials.



MODULE DESCRIPTION

TRANSPORTATION MODULE

WEATHER MODULE

GEOLOACATION AND MAP MODULE

CONVERTOR MODULE

SEARCH MODULE

C:\project\review3.pyw - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

review3.pyw

```python
from tkinter import *
from tkinter import ttk
import wikipedia
from geopy.geocoders import Nominatim
from googletrans import Translator
import gmplot
import os
from geopy import distance
import requests
from bs4 import BeautifulSoup
import requests, json
import datetime
from suntime import Sun
from timezonefinder import TimezoneFinder
root=Tk()
root.title("Escort Amigo")
root.iconbitmap("C:/project/i1.ico")

r=IntVar()
r.set(1)
frame=LabelFrame(root,text="Options Available",padx=50,pady=50)#padding inside the frame
frame.pack(padx=40,pady=40)#padding outside of frame

options=[("Geo Locator",1),("Weather",2),("Map",3),("Train TT",4),("Convertor",5),("Search",6)]

for text,num in options:
    Radiobutton(frame,text=text,variable=r,value=num).pack(anchor=W)

def click(value):
    if value==1:
        top=Toplevel()
        top.title("Geo Locator")
        top.iconbitmap("C:/project/i1.ico")
        global options1
        t=IntVar()
        t.set(1)
        frame=LabelFrame(top,text="Find",padx=50,pady=50)#padding inside the frame
        frame.pack(padx=10,pady=10)#padding outside of frame

        options1=[("Place through Zipcode",1),("Zipcode through place",2),("Co-ordinates through place",3),("Place through co-ordinates",4),("Distance Between

        for text,num in options1:
```

Line 1, Column 1 Tab Size: 4 Python

Code Snippets with their respective outputs

```python
#Geographical coorditnates finder
# importing geopy library
from geopy.geocoders import Nominatim
from geopy.distance import great_circle

# calling the Nominatim tool
geolocator = Nominatim(user_agent="geoapiExercises")
ladd1 = input("Enter First Place Name: ")
location1 = geolocator.geocode(ladd1)
ladd2 = input("Enter Second Place Name: ")
location2 = geolocator.geocode(ladd2)
print("Distance between places: ",great_circle(((location1.latitude, location1.longitude)), (location2.latitude, location2.longitude)).km,"Km")
```

```
... Enter First Place Name:  Delhi
Enter Second Place Name:  Kolkata
Distance between places:  1306.1234240366712 Km
```

```python
#map of a place
import gmplot
import os
from geopy.geocoders import Nominatim
loc = Nominatim(user_agent="GetLoc")
place=input("Enter place name: ")
getLoc = loc.geocode(place)
lat=getLoc.latitude
long=getLoc.longitude
gmap1 = gmplot.GoogleMapPlotter(lat,long, 13 )
cpath="C:\\Users\\shubh\\Desktop\\map11.html"
gmap1.draw(cpath)
os.startfile(cpath)
```

```
...  Enter place name:  Sagar Madhya Pradesh
```

file:///D:/map11.html

```python
#Language converter
from googletrans import Translator

#initialize the Translator
translator = Translator()

text = input("Enter your Text: ")
source_lan = input("Translate from: ")
translated_to= input("Translate to: ")

#translate text
translated_text = translator.translate(text, src=source_lan, dest = translated_to)

print(f"The Actual Text was {text}")
print(f"The Translated Text is: {translated_text.text}")
```

```
Enter your Text:  I love my country
Translate from:  English
Translate to:  Hindi
The Actual Text was I love my country
The Translated Text is: मुझे अपने देश से प्यार है
```

```python
#Currency Converter
import requests
class RealTimeCurrencyConverter():
    def __init__(self,url):
        self.data= requests.get(url).json()
        self.currencies = self.data['rates']
    def convert(self, from_currency, to_currency, amount):
        initial_amount = amount
    #first convert it into USD if it is not in USD.
    # because our base currency is USD
        if from_currency != 'USD' :
            amount = amount / self.currencies[from_currency]

    # limiting the precision to 4 decimal places
        amount = round(amount * self.currencies[to_currency], 4)
        return amount
currfrom = input("Enter Initial Currency: ")
currto = input("Enter Final Currency: ")
amt = int(input("Enter Amount to Convert: "))
url = 'https://api.exchangerate-api.com/v4/latest/USD'
converter = RealTimeCurrencyConverter(url)
print("Converted Amount= ",converter.convert(currfrom,currto,amt))
```

Python

```
Enter Initial Currency:  EUR
Enter Final Currency:  INR
Enter Amount to Convert:  1
Converted Amount=  86.0441
```

```python
import requests
from bs4 import BeautifulSoup

# user define function
# Scrape the data
def getdata(url):
    r = requests.get(url)
    return r.text


# input by geek
from_Station_code = input("Enter First Station Code: ")
from_Station_name = input("Enter First Station Name: ")
To_station_code = input("Enter Second Station Code: ")
To_station_name = input("Enter Second Station Name: ")
date=input("Enter date of journey(dd-mm-yyyy): ")
# url
url = "https://www.railyatri.in/booking/trains-between-stations?from_code="+from_Station_code+"&from_name="+
        from_Station_name+"+JN+&journey_date="+date+"&src=tbs&to_code="+To_station_code+"&to_name="+To_station_name+"
         +JN+&user_id=-1636260722&user_token=61636260722&utm_source=train_ticket_dweb_search"

# pass the url
# into getdata function
htmldata = getdata(url)
soup = BeautifulSoup(htmldata, 'html.parser')
# find the Html tag
# with find()
# and convert into string
data_str = ""
data_str1= ""
for item in soup.find_all("div", class_="TravelTimeInfo"):
    data_str = data_str + item.get_text()
result = data_str.split("\n")
for item in soup.find_all("div", class_="namePart"):
    data_str1 = data_str1 + item.get_text()
```

```python
result = data_str.split("\n")
for item in soup.find_all("div", class_="namePart"):
    data_str1 = data_str1 + item.get_text()
result1 = data_str1.split("\n")


print("Train between "+from_Station_name+" and "+To_station_name)
print("")

result=list(filter(None, result))
#print(result)
result1=list(filter(None,result1))
#print(result1)
print('\n')
# Display the result

for i in result1:
    print(i)
    print('\n')
    for j in result:
        if j=='                                    ':
            for k in range(result.index(j)):
                print(result[k])
            k=result.index(j)+1
            print('\n')
            del result[0:6]
            break
```

```
Enter First Station Code:  GWL
Enter First Station Name:  Gwalior
Enter Second Station Code:  BPL
Enter Second Station Name:  Bhopal
Enter date of journey(dd-mm-yyyy):  11-11-2021
Train between Gwalior and Bhopal
```

```
02806  NEW DELHI - VISAKHAPATNAM Andhra Pradesh SF Special


23:45GWL
 GWALIOR JN
4 H 45 M
04:30 BPL
BHOPAL JN


02173  HAZRAT NIZAMUDDIN - JABALPUR Shridham SF Special


19:03GWL
 GWALIOR JN
6 H 52 M
01:55 BPL
BHOPAL JN


06250  HAZRAT NIZAMUDDIN - YESVANTPUR Sampark Kranti SF Speci
```

show more (open the raw output data in a text editor) ...

```
06216  NZM MYS SPL
```

```python
#address from zipcode
# Importing required module
from geopy.geocoders import Nominatim

# Using Nominatim Api
geolocator = Nominatim(user_agent="geoapiExercises")

# Zipocde input
zipcode = input("Enter zipcode of place to search: ")

# Using geocode()
location = geolocator.geocode(zipcode)

# Displaying address details
print("Zipcode:",zipcode)
print("Details of the Zipcode:")
print(location)
```

```
Enter zipcode of place to search:  221001
Zipcode: 221001
Details of the Zipcode:
Sadar, Varanasi, Uttar Pradesh, 221001, India
```

```python
#Zipcode finder
# import module
from geopy.geocoders import Nominatim

# initialize Nominatim API
geolocator = Nominatim(user_agent="geoapiExercises")

# place input by geek
place = input("Enter name of place: ")
location = geolocator.geocode(place)

# traverse the data
data = location.raw
loc_data = data['display_name'].split()
#print("Full Location")
#print(loc_data)
print("Zip code : ",loc_data[-2])
```

```
Enter name of place:  Sigra varanasi
Zip code :  221010,
```

```python
#Place finder using latitude and longitude
# importing modules
from geopy.geocoders import Nominatim

# calling the nominatim tool
geoLoc = Nominatim(user_agent="GetLoc")
latlong=input("Enter Latitude and Longitude of place: ")
# passing the coordinates
locname = geoLoc.reverse(latlong)

# printing the address/location name
print(locname.address)
```

```
Enter Latitude and Longitude of place:  26.7674446, 81.109758
Gosainganj, Mohanlalganj, Lucknow, Uttar Pradesh, 227125, India
```

Place finder using latitude and longitude

Place information finder

```python
import wikipedia
query= input("Enter the Place name to search:")
print("Information found:")
results = wikipedia.summary(query, sentences=4)
print(results)
```

Python

```
Enter the Place name to search:London
Information found:
London is the capital and largest city of England and the United Kingdom. It stands on the River Thames in south-east England at the head of a 50-mile (80 km) estuary down to the North Sea, and has been a major settlement for two millennia. The City of London, its ancient core and financial centre, was founded by the Romans as Londinium and retains boundaries close to its medieval ones. Since the 19th century, "London" has also referred to the metropolis around this core, historically split between the counties of Middlesex, Essex, Surrey, Kent, and Hertfordshire, which largely makes up Greater London, the region governed by the Greater London Authority.
```

Escort Amigo

File Home Share View

This PC > WINDOWS (C:) > project > Escort Amigo

Search Escort Amigo

Quick access
- Desktop
- Downloads
- Documents
- Pictures
- This PC
- gui
- project
- Screenshots
- The.Witcher.S02.1080p.N

OneDrive - Personal

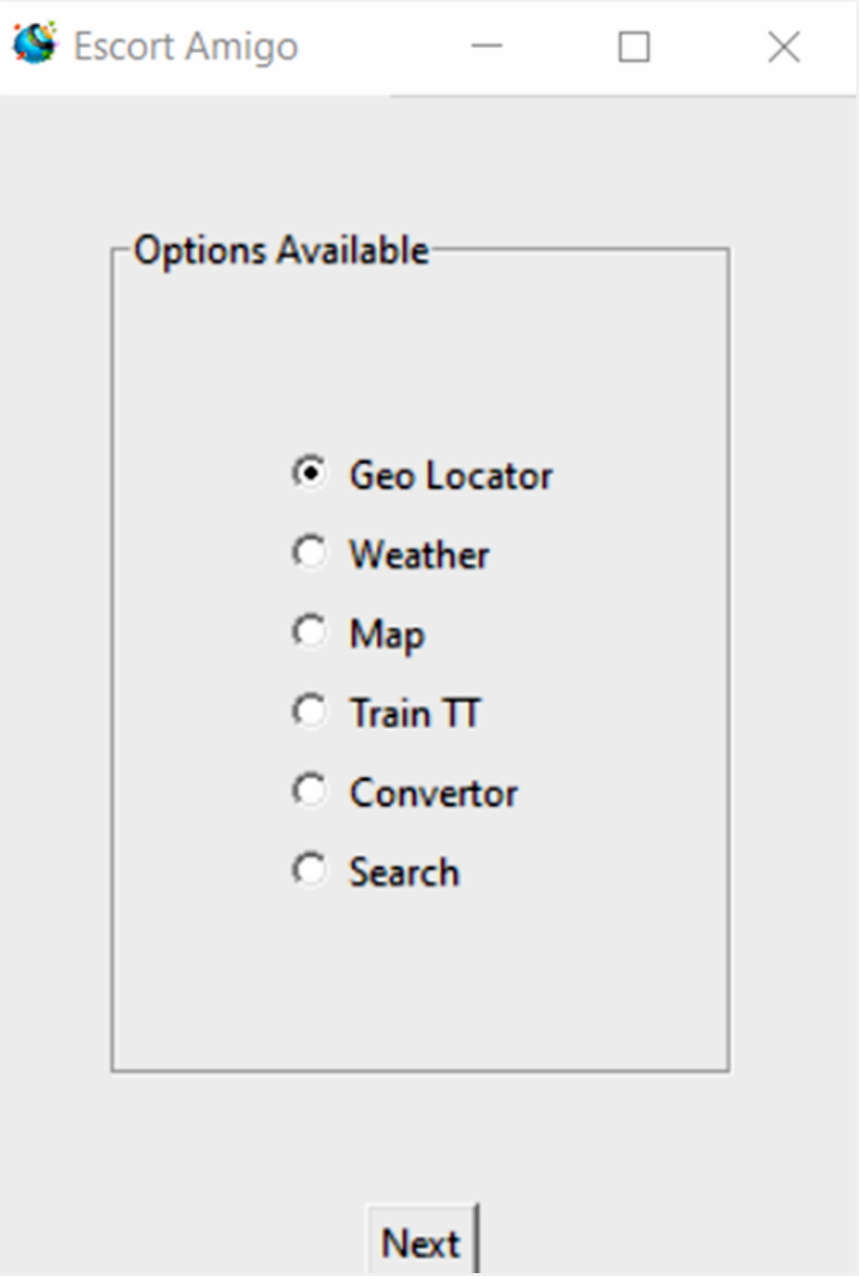
This PC
- 3D Objects
- Desktop
- Documents
- Downloads
- Music
- Pictures
- Videos
- WINDOWS (C:)
- Softs (D:)
- Miscellaneous (E:)
- Trash (F:)

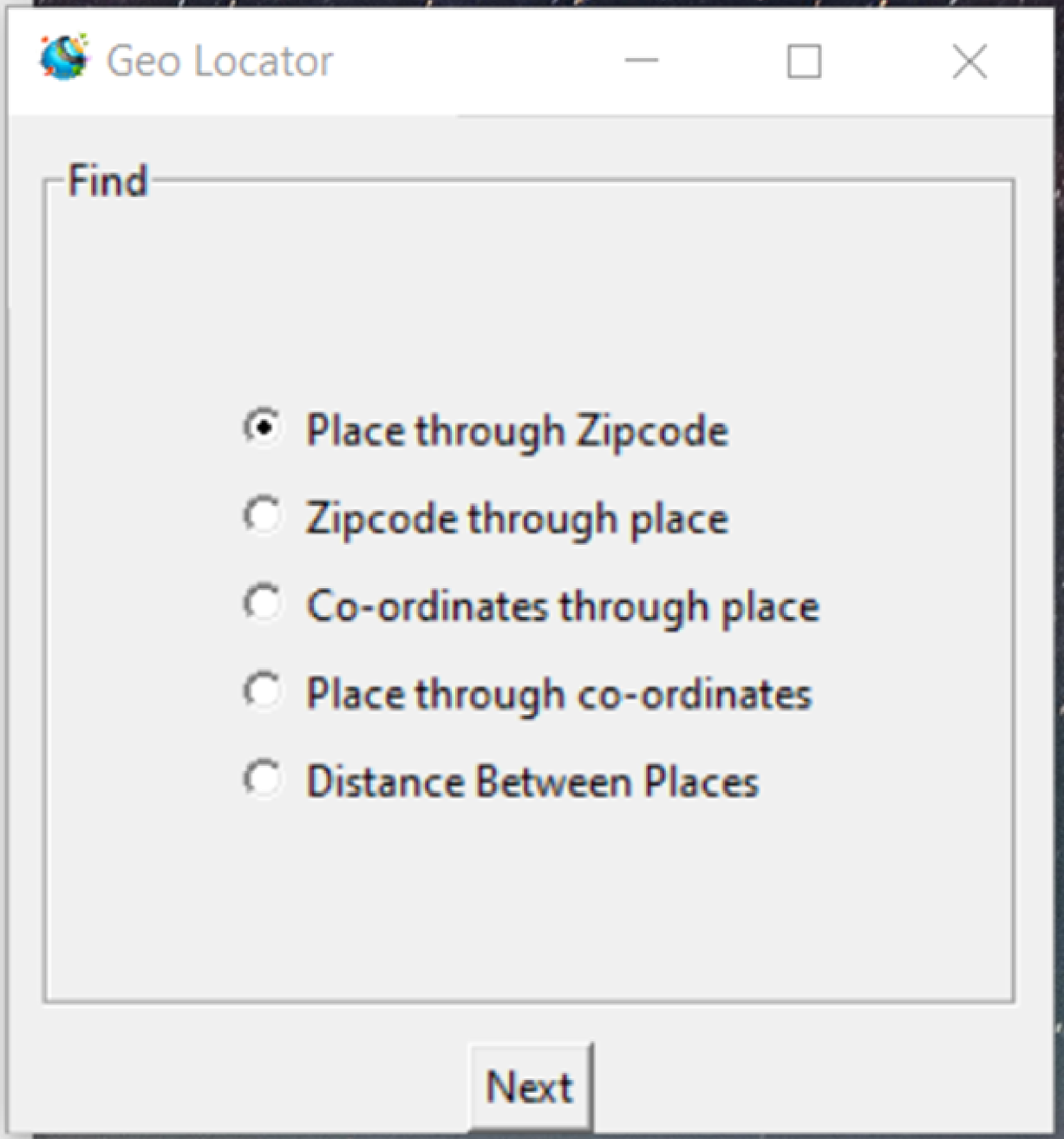
Network

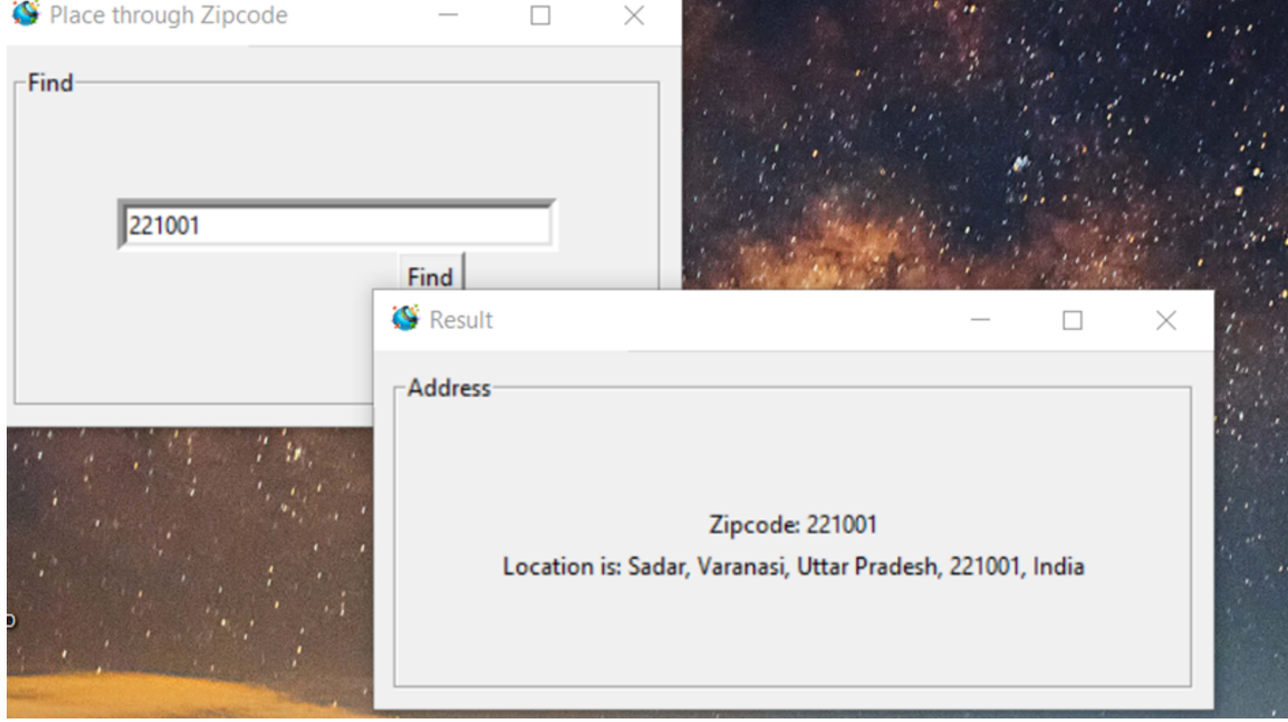
Escort Amigo.exe

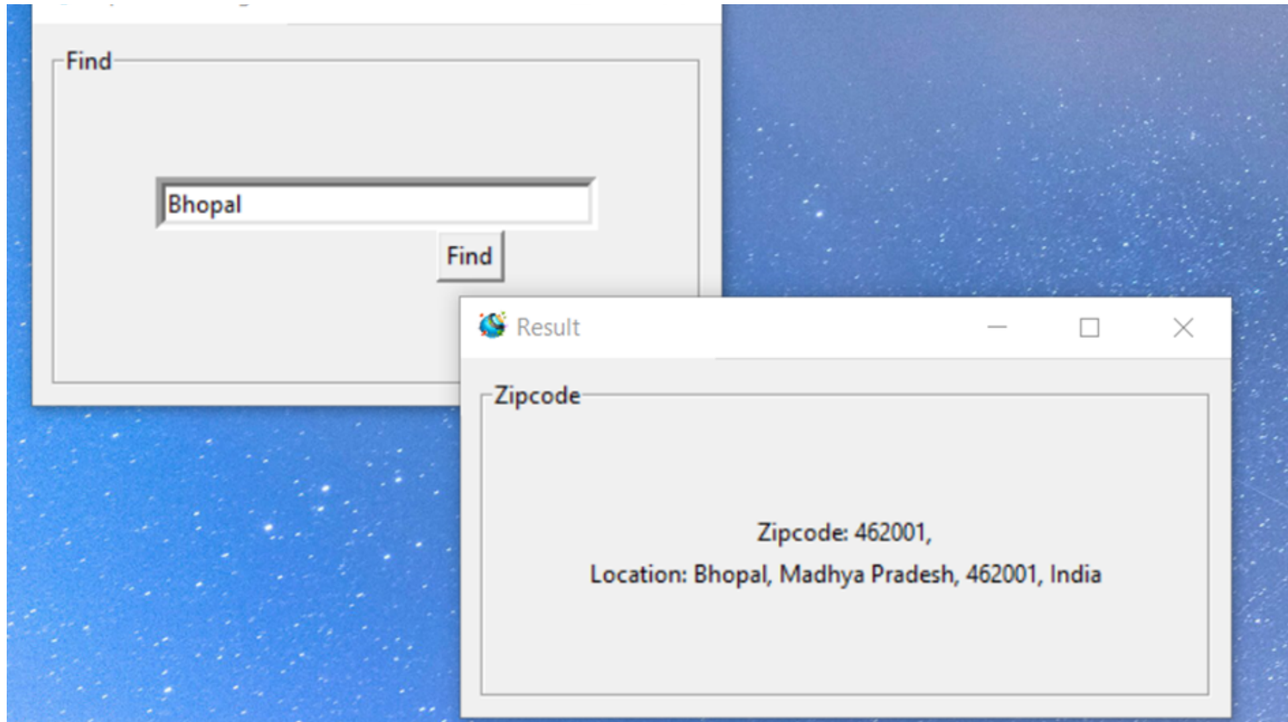
1 item

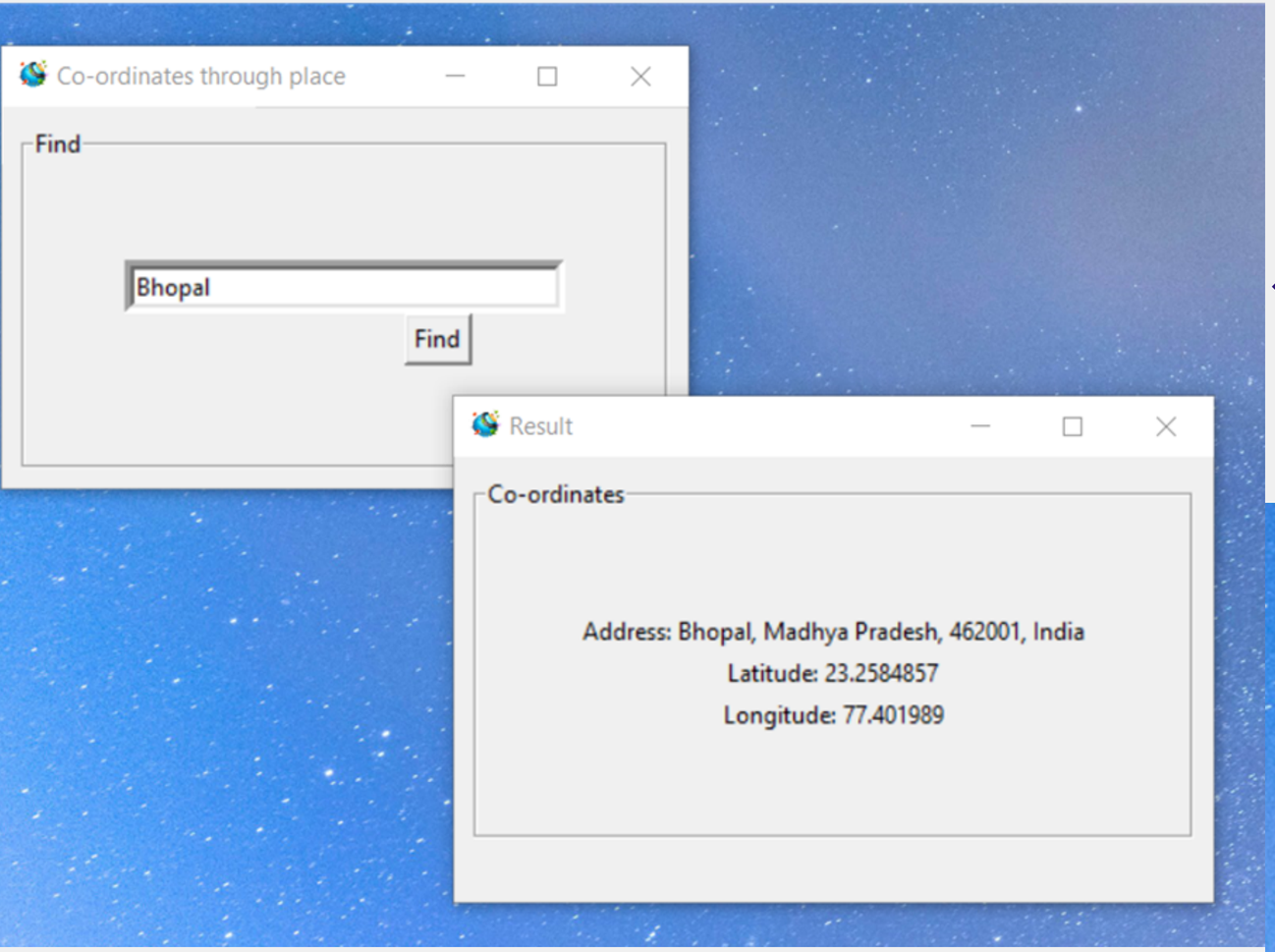
TESTING AND RESULTS



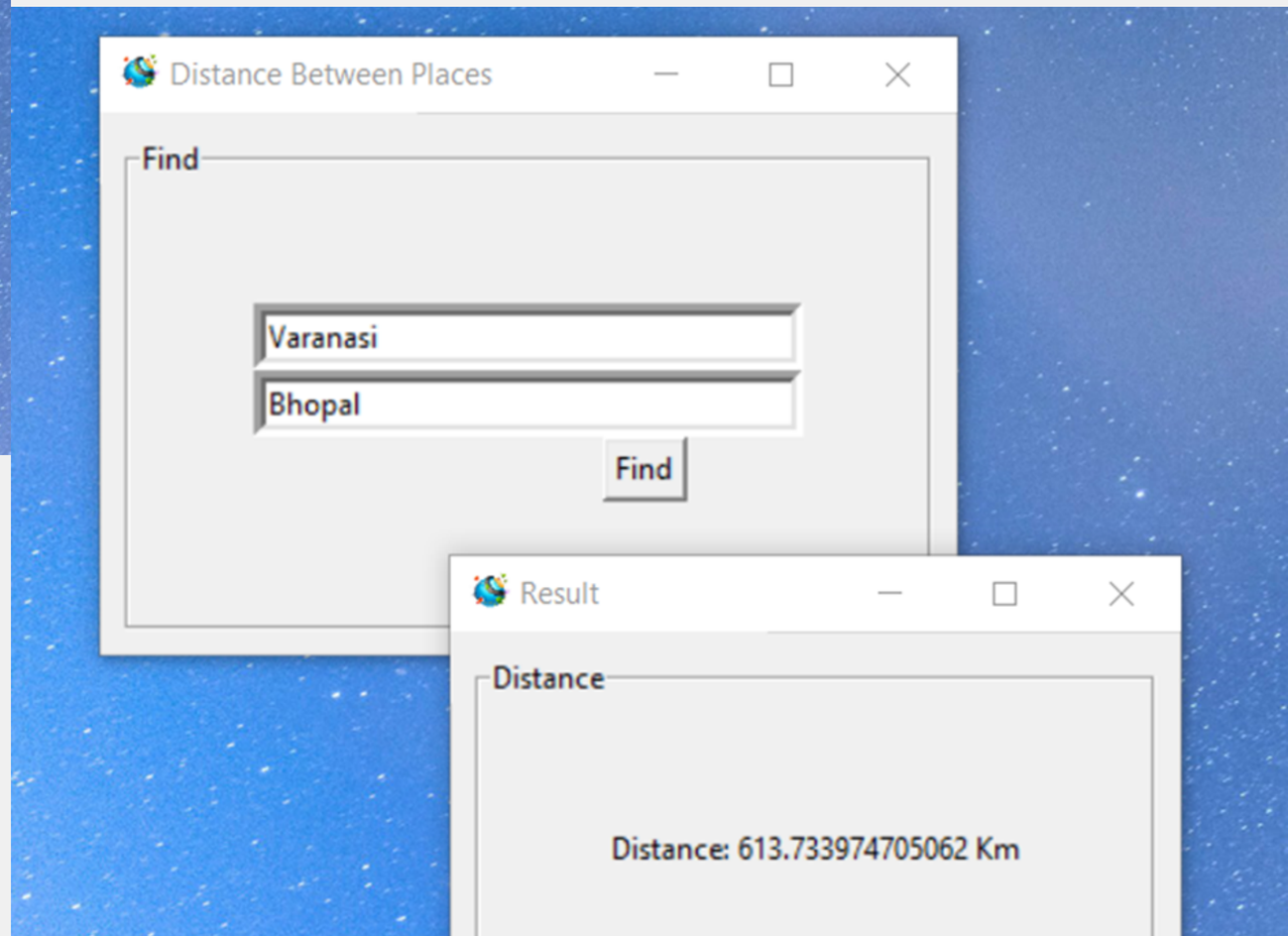

Geo Locator
Find
Place through Zipcode
Zipcode through place
Co-ordinates through place
Place through co-ordinates
Distance Between Places
Next


Place through Zipcode
Find
221001
Find
Result
Address
Zipcode: 221001
Location is: Sadar, Varanasi, Uttar Pradesh, 221001, India

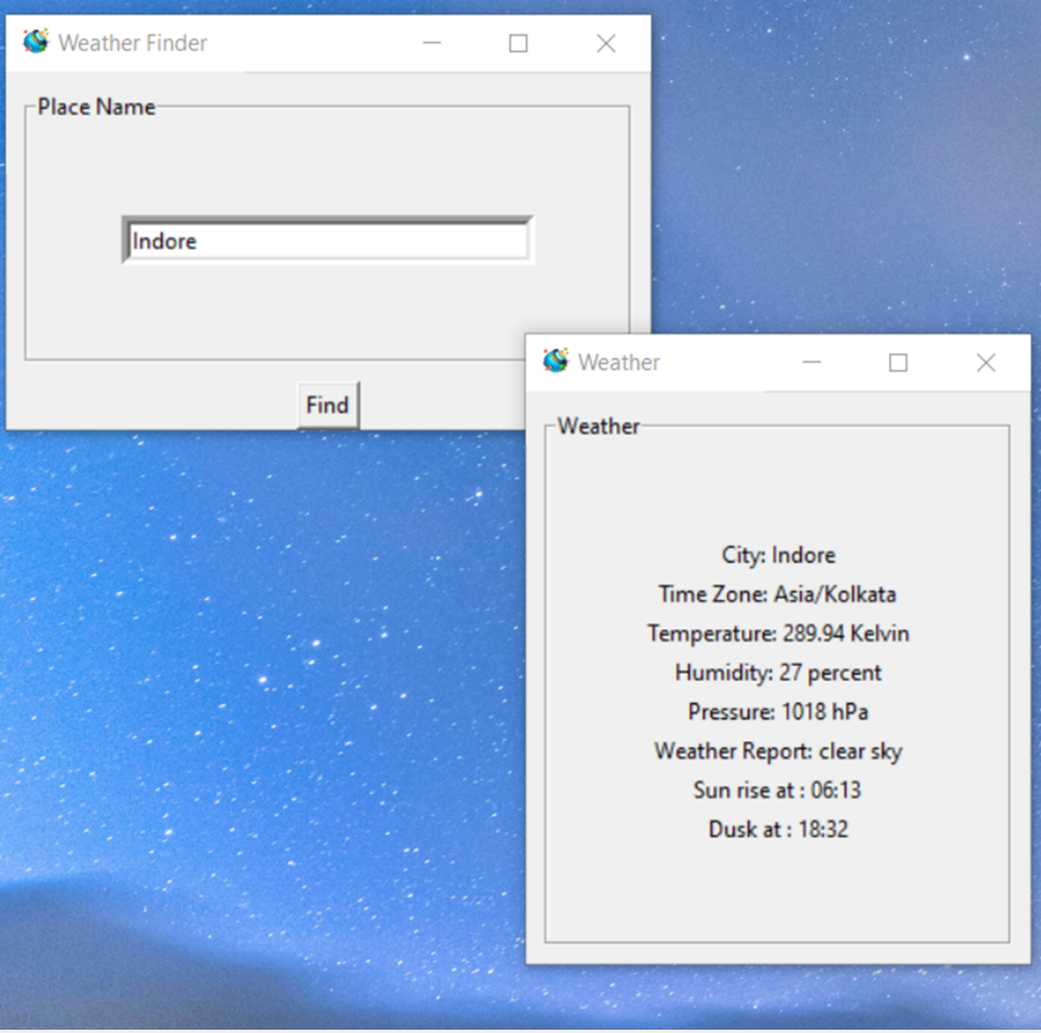

Find
Bhopal
Find
Result
Zipcode
Zipcode: 462001,
Location: Bhopal, Madhya Pradesh, 462001, India

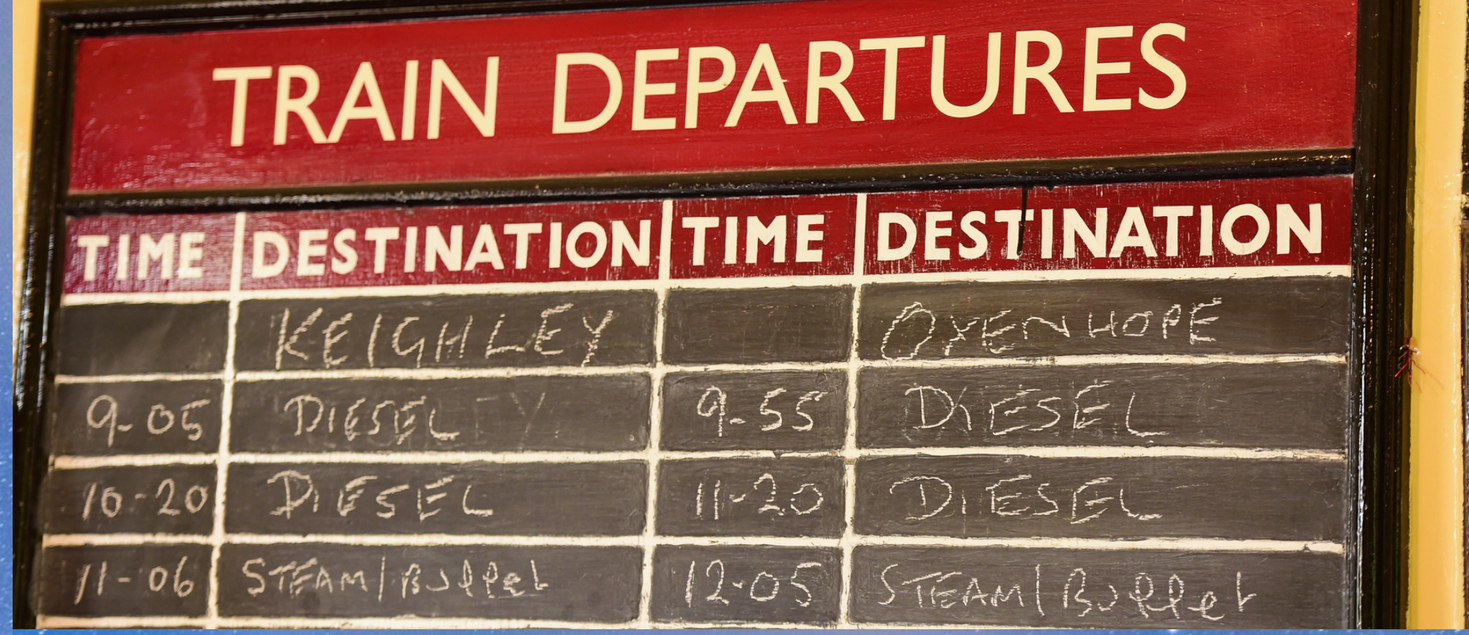


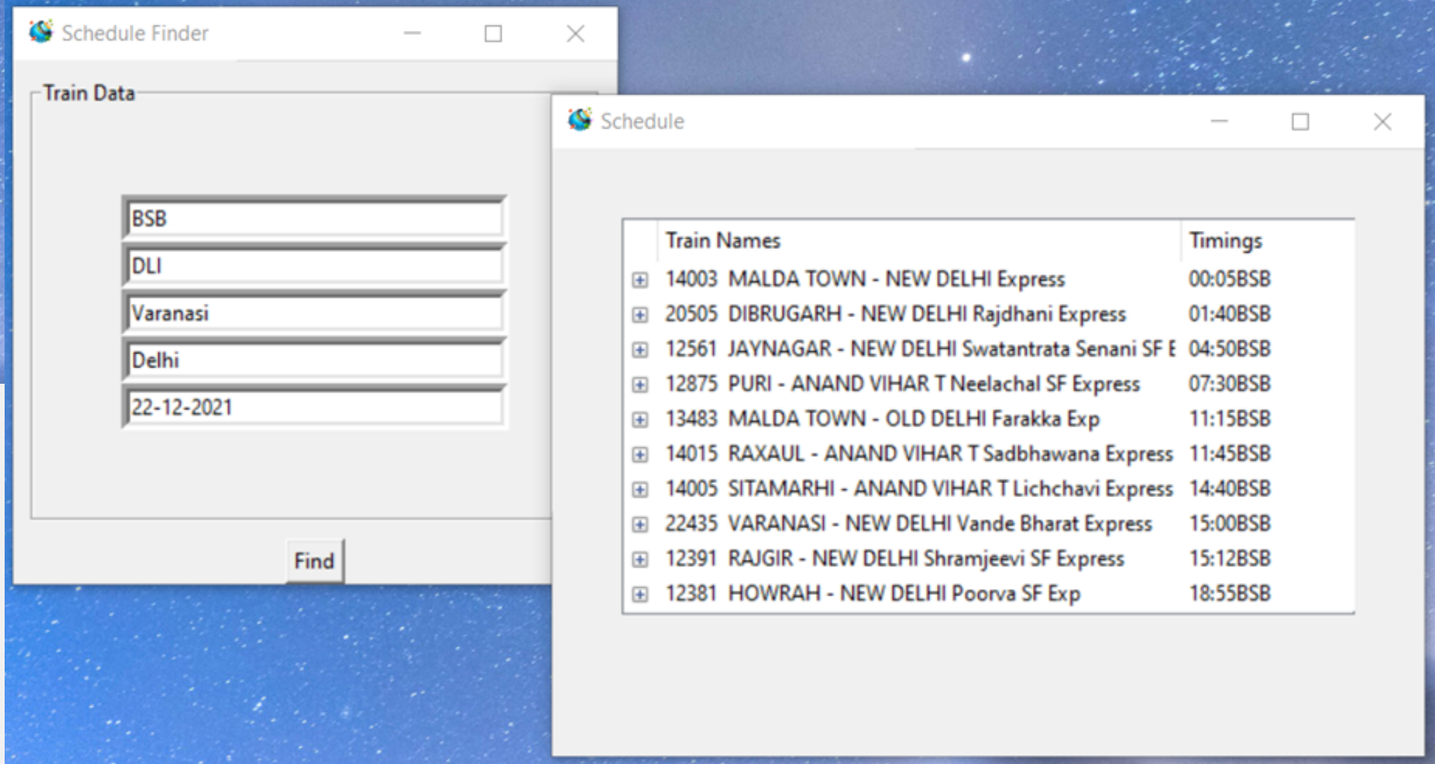
<-- Co-ordination through place



Distance between places -->







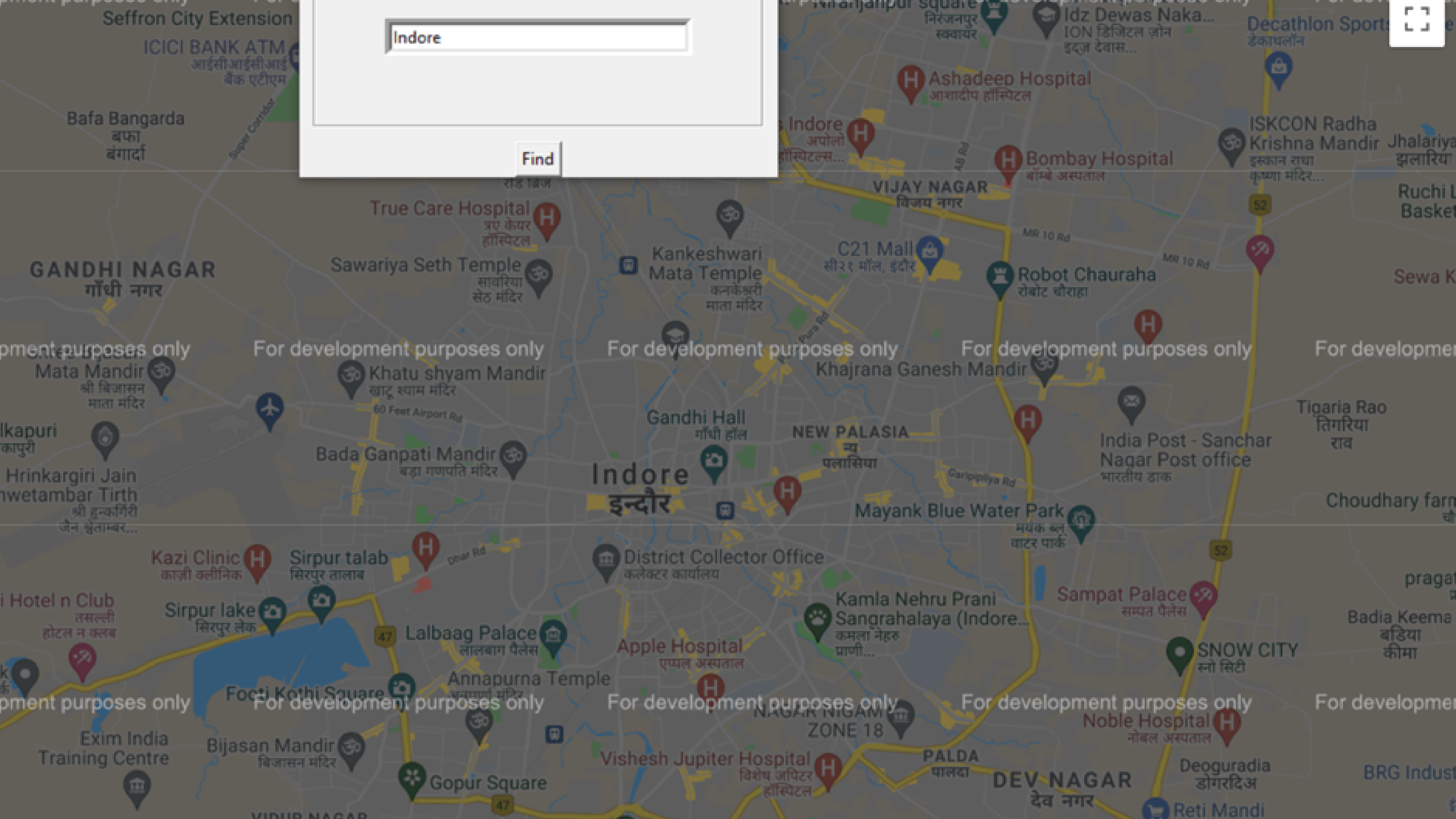
↑

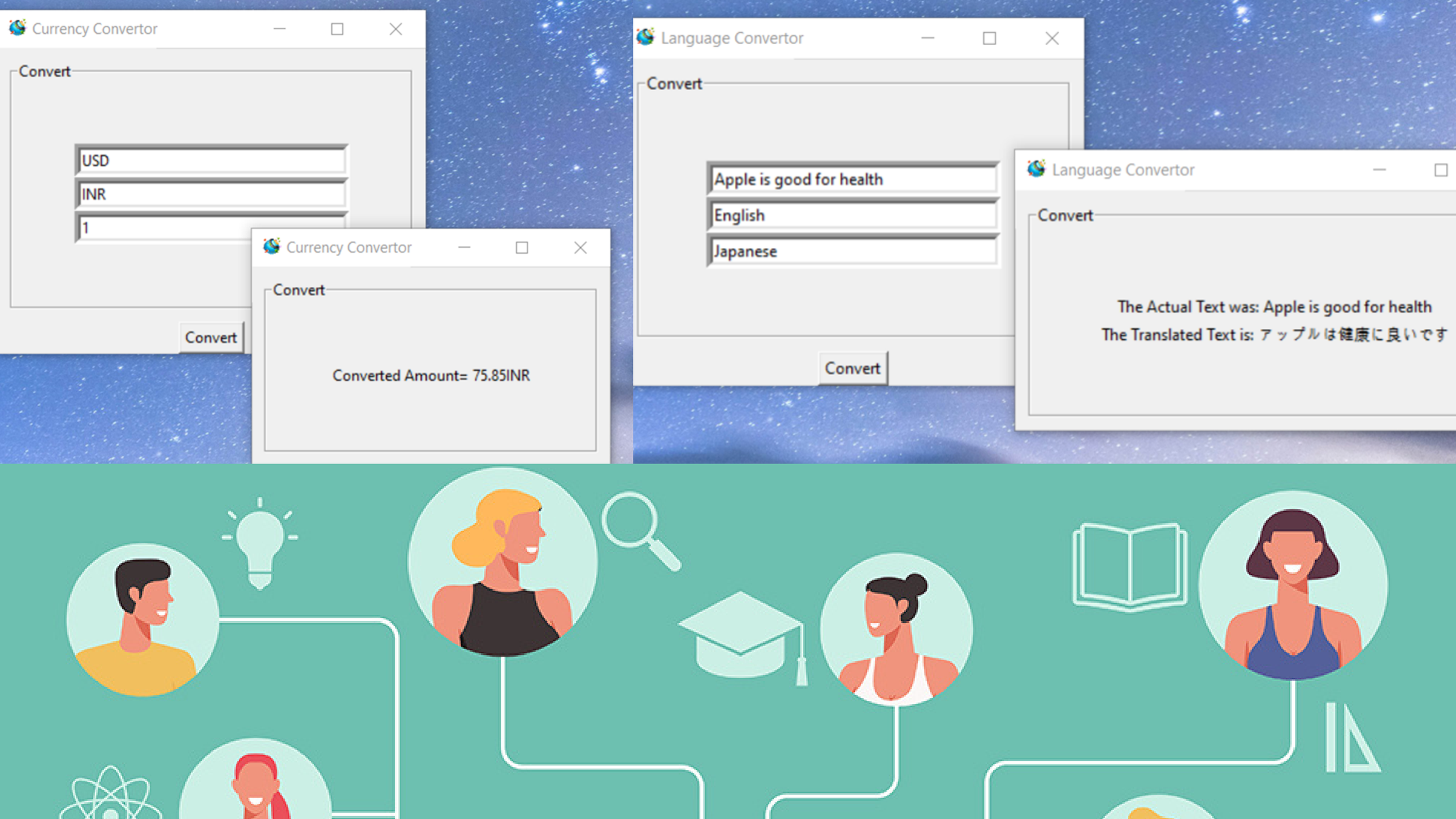
Weather Finder

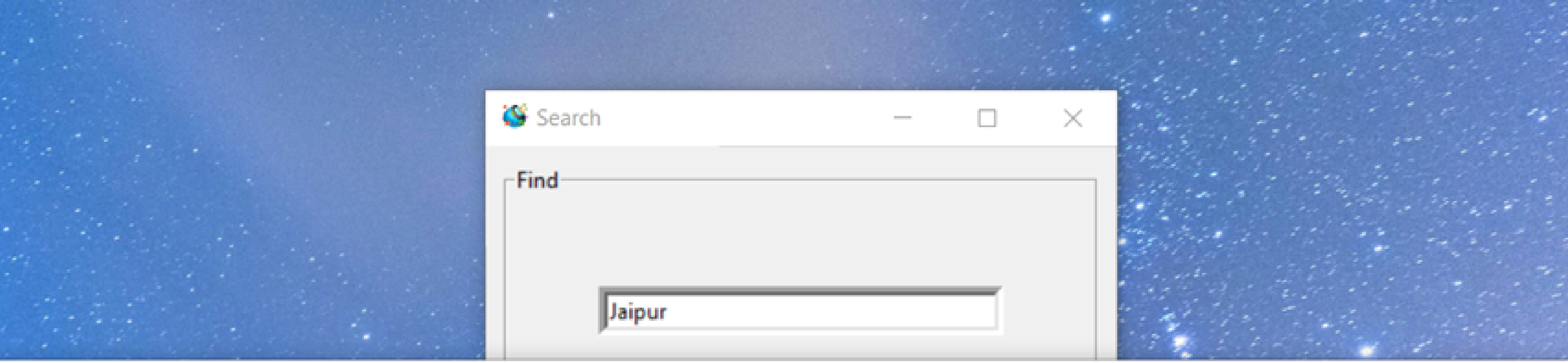
Schedule Finder -->

Indore

Find


Seffron City Extension
ICICI BANK ATM
Bafa Bangarda
बफा
बंगार्दा
True Care Hospital
Sawariya Seth Temple
GANDHI NAGAR
गाँधी नगर
Kankeshwari Mata Temple
C21 Mall
Robot Chauraha
VIJAY NAGAR
Bombay Hospital
Ashadeep Hospital
Idz Dewas Naka...
Decathlon Sports
ISKCON Radha Krishna Mandir
Jhalariya
Ruchi Basket
Sewa K
For development purposes only
Khatu shyam Mandir
Khajrana Ganesh Mandir
Gandhi Hall
NEW PALASIA
India Post - Sanchar Nagar Post office
Tigaria Rao
Bada Ganpati Mandir
Indore
इन्दौर
Mayank Blue Water Park
Choudhary farm
Kazi Clinic
Sirpur talab
District Collector Office
Sirpur lake
Lalbaag Palace
Apple Hospital
Kamla Nehru Prani Sangrahalaya (Indore...
Sampat Palace
SNOW CITY
Badia Keema
Annapurna Temple
Focti Kothi Square
Exim India Training Centre
Bijasan Mandir
Gopur Square
Vishesh Jupiter Hospital
NAGAR NIGAM ZONE 18
PALDA
DEV NAGAR
Noble Hospital
Deoguradia
Reti Mandi
BRG Indust


Currency Convertor
Convert
USD
INR
1
Convert
Currency Convertor
Convert
Converted Amount= 75.85INR
Language Convertor
Convert
Apple is good for health
English
Japanese
Convert
Language Convertor
Convert
The Actual Text was: Apple is good for health
The Translated Text is: アップルは健康に良いです



Search Result

formation Found

Jaipur ((listen); Hindi: जयपुर; IAST: jayapura) is the capital and the largest city of the Indian state of Rajasthan. As of 2011, the city had a population of 3.1 million, making it the tenth most populous city in the country. Jaipur is also known as the Pink City, due to the dominant colour scheme of its buildings. It is located 268 km (167 miles) from the national capital New Delhi. On 6 July 2019, UNESCO World Heritage Committee inscribed Jaipur the "Pink City of India" among its World Heritage Sites. Jaipur was founded in 1727 by the Kacchawa Rajput ruler Jai Singh II, the ruler of Amer, after whom the city is named.

CONCLUSION

In this project, we presented a python based software that gives you all the information about a place that users want to visit on their vacations. This application helps the users to save time and get the information they asked for, no additional information such as pop-up ads, redirects, or information regarding some other place, etc. It is a user-friendly application.

Thank you