**Title: Case Study on Hyperloop Network Operation**

**Contents**

Discussion..………………………………………………………………………………………..3

* RQ 4-Pseudo Code .………………………………………………………………………3
* RQ 5- Application Design and Function and modules…….………………………………3
* Main()……………………………………………………………………………..3
* Gmap1() .. …………………………………………………………………………3
* Speed()…………………………………………………………………………….4
* Tur\_layout()……………………………………………………………………….5
* drawbar()…………………………………………………………………………..5

RQ 6- System testing with an Example…………………………………………………………...5

RQ1-Code…………………………………………………………………………………………8

RQ1-Output….……………………………………………………………………………………8

RQ2-Code…………………………………………………………………………………………9

RQ2-Output………………………………………………………………………………………10

RQ3-Code………………………………………………………………………………………..10

RQ3-Output………………………………………………………………………………………10

RQ7-Code………………………………………………………………………………………..11

RQ7-Output………………………………………………………………………………………12

RQ8-Code………………………………………………………………………………………..12

RQ8-Output………………………………………………………………………………………13

RQ9-Code………………………………………………………………………………………..14

RQ9-Output………………………………………………………………………………………14

RQ10-Code………………………………………………………………………………………15

RQ10-Output……………………………………………………………………………………..15

References………………………………………………………………………………………..16

**RE 04**

**Pseudo Code:**

Import all library NumPy, Pandas, Matplotlib, Geopy, Turtle and plotly.

Collect data of 10 different city’s and Create a Data frame and Make REA.csv file.

Create five Function drawbar, tur, speed,asd,gmap1

Create drowbar for create a turtle horizontal bar plot.

Make graph using forword and left right class and create a horizontal bar plot.

Create tur function to make turtle GUI using screen class and passes the values in drowbar function.

Tur function also generate a horizontal barplot using matplotlib and pandas using plt.brah() and df.plot.barh classes.

Tur function also shows which function is better for the transportation.

Create speed function to check the time used in each transportation.

Create function calls tur function and provides all the spent time values.

Create main function to start the complete program.

Main function take input using pandas read\_csv class and import Rea.csv file .

using input class take two city input.

Calculate the lag long using geopy library.

using geodesic of geopy calculate the distance between both locations.

Check the route if route is conformed then execute gmap1 function and send the lat long of both cities.

Create a gmap1 function, this function use for visualization purpose.

Gmap1 function locate both country on world map using plotly library and show the path between both city using go class.

**RE 05**

**Application Design and functions and modules:**

Travel times are approximated and incorporate security check, ticketing, refuel, and sitting tight for flight. For the most precise outcomes, inside and out practicality examines must be executed. To take into account most extreme course associations, In this project we have made the accompanying suppositions:

1) Virgin Hyperloop One paces are 1080 km/h

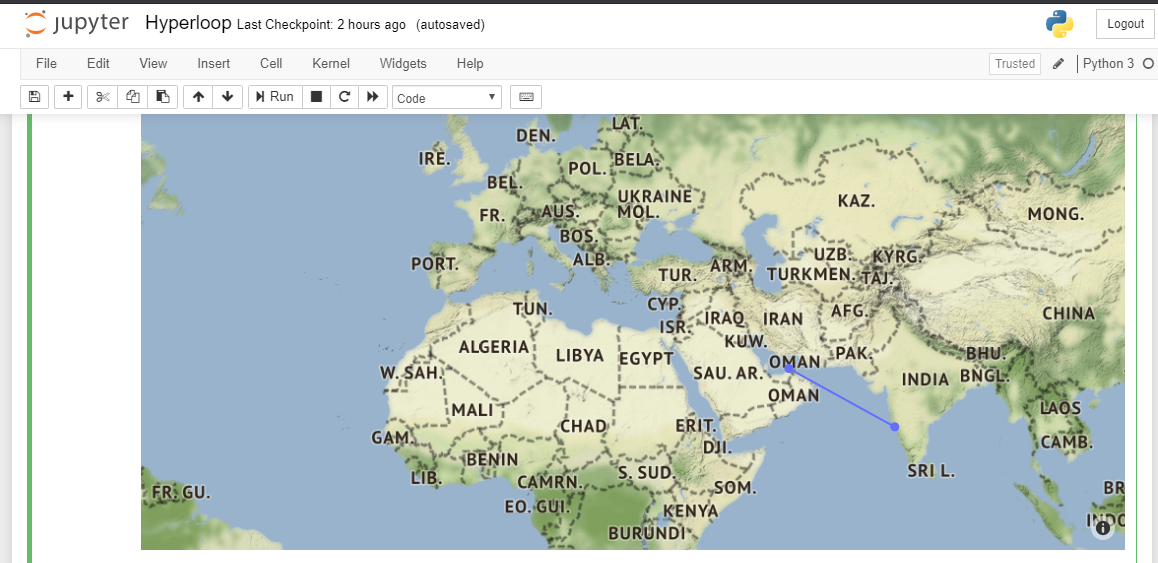
2) Routes are a straight excursion (point to point), paying little mind to territory (water, mountains)

3) If a method of transportation is as of now inaccessible for a course, we will assess the ideal opportunity for correlation.

In This project Main( ) function is the main function of this project when we execute

**1.** **Main (): -** Function then this function imports all the library that is used in this project (NumPy, pandas, plotly, pandas, matplotlib, geopy). the using panda’s library classes we import the REA.csv file or information of ten city. Then we take two city names as an input in city andcity1 variable. Using the geopy library classes we calculate the latitude and longitude of both cities. Then we calculate we calculate the distance between both cities using geopy geodesic library. And the gmap1 function executed.

**2. Gmap1():-** This function is use for the visualization purpose.in this function we import plotly library and make a world base map .using the world base map we visualize a map and location and path between both city as show in fig.



**Fig: Distance visualization between Source to Designation**

**3. Speed (): -** This function is basically calculated distance per minute according to the different different transportation methods. This function helps to calculate the travel time between both cities. Calculate the traveling time between both country and store in a list that may help to visualization the graphs. After all the calculation the function executed tur\_layout () function.

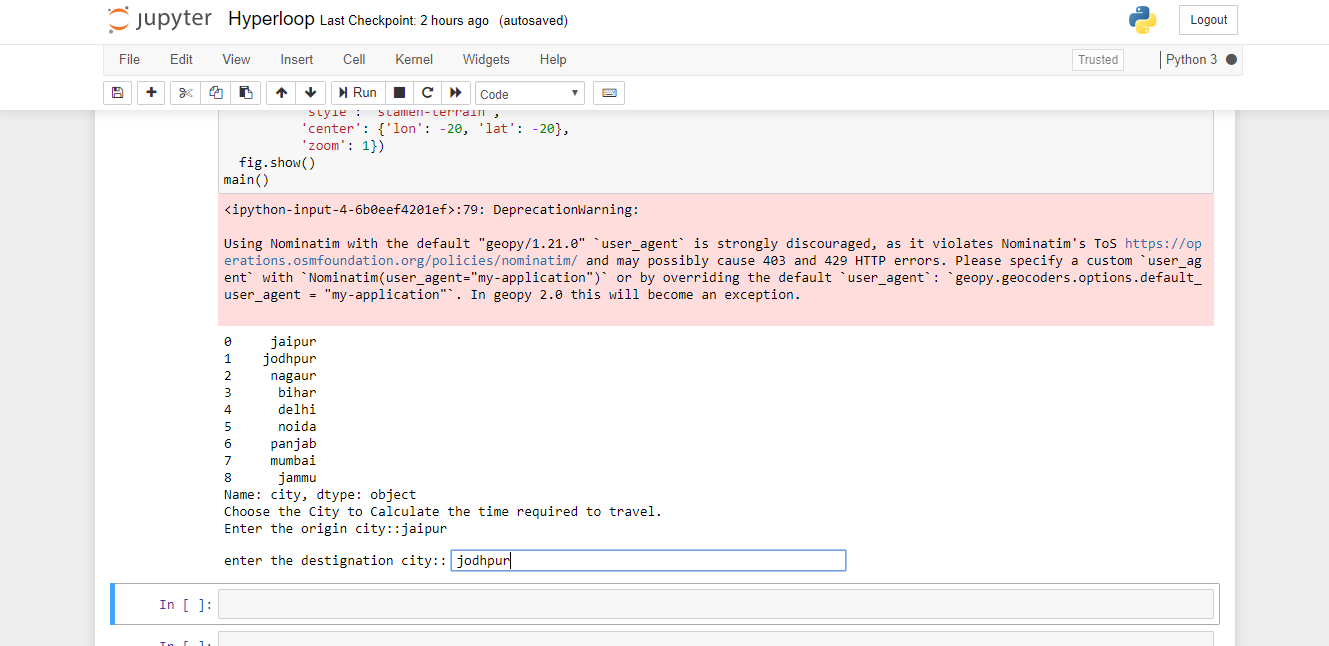
**4. Tur\_layout ( ):-** this function is parent function of drawBar () function in Tur\_layout ( ) bar function we initialize the turtle library and perform iteration on drawbar() function and pass all the list values of lists. This function also visualize the graph using pandas and matplotlib library classes. Using plt.brah() and pd.plot.brah() classes we visualize the horizontal plot using pandas and matplotlib.

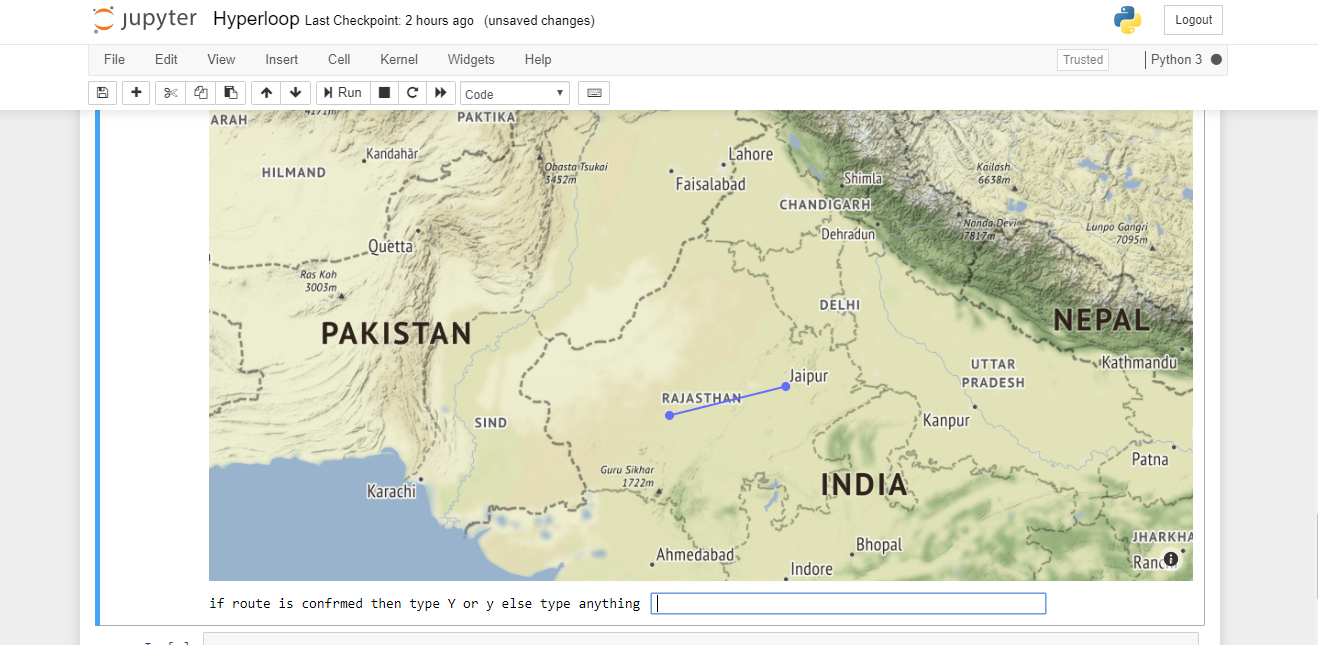
**5. drawBar (): -** this is child function of the tur () function in this function we make a turtle graph using different different classes. In this function we use left right and forward class of turtle library.

**RQ 6**

**System testing with an Example:**

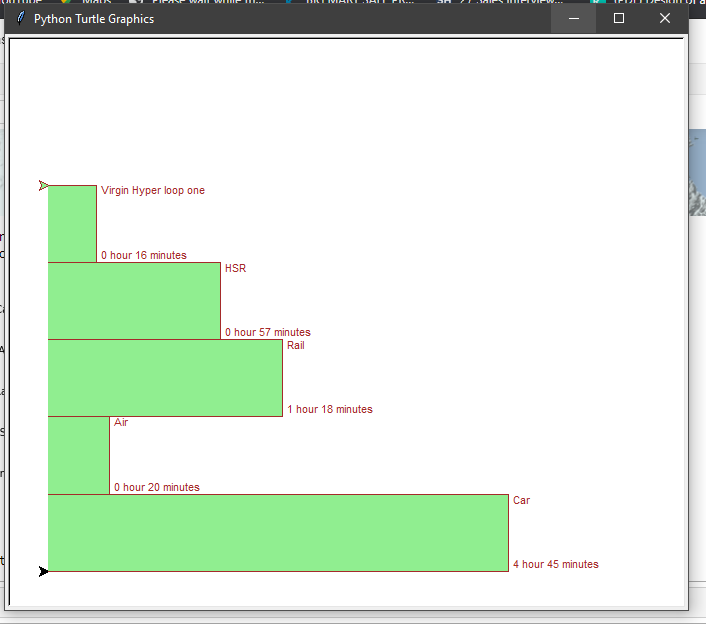
For this project execution we required a jyputer-notebook with high configuration system and 8 gb Mam. Jyputer notebook required to follow all the prerequisites mentioned in requirement.txt.

Example: when we run we input first city Jaipur and type enter then second city as show in fig.After that we type y for the confirmation of route estimation. Then it shows the world map with lag long as show in fig.



**Fig: Visualization between two cities using Basemap**

Then it shows three graph one is turtle and another two using pandas and matplotlib as show in fig



**Fig: Graph of Transportation system vs Traveling Time(Using Turtle library)**

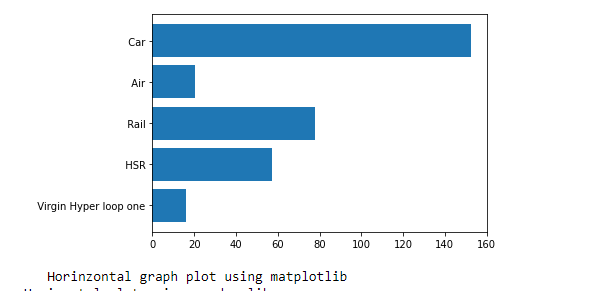


Fig: Graph of Transportation system vs Traveling Time(Using Matplotlib library)

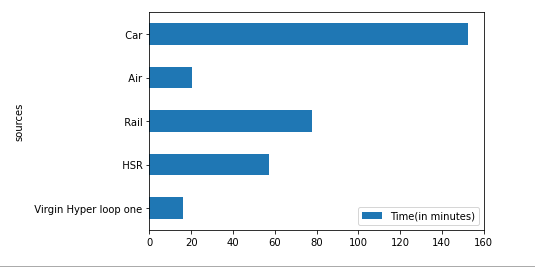
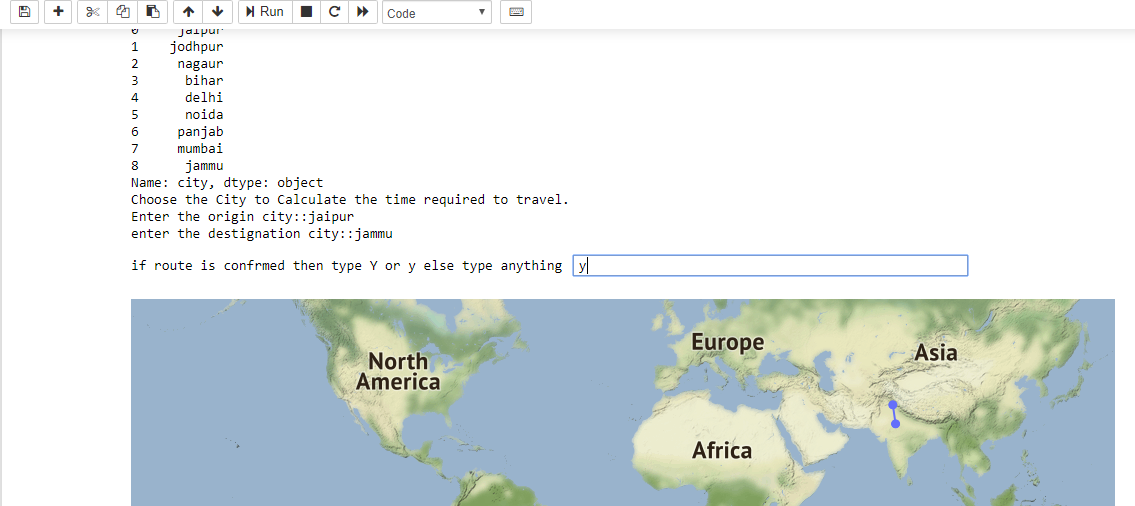


Fig: Graph of Transportation system vs Traveling Time(Using Pandas library)

**Data Preparation:**



**Program Code:**

**Import Libraries:**

import turtle

import matplotlib.pyplot as plt

import numpy as np

import pandas as pd

from geopy.geocoders import Nominatim

from geopy.distance import great\_circle

from geopy.distance import geodesic

import pandas as pd

import plotly.graph\_objects as go

**RQ1-Code:**

def main():

df=pd.read\_csv("city1.csv") #Read city1.csv file data from Pandas class.

print(df['city'][0:9])

print("Choose the City to Calculate the time required to travel.")

geolocator = Nominatim(timeout=4)

city1=input('Enter the origin city::') # take Source city name.

city =input('enter the destignation city::') #Take destignation city name.

if city1.lower()=='na' or city.lower()=='na':#Check the city name is vaiid or not.

print('invalid input city name')

exit

else:

loc1 = geolocator.geocode(city1) # Geocode class is use to calculate the cordinates of the city.

loc = geolocator.geocode(city)

lat=[loc1.latitude,loc.latitude] #loc1 or loc is object of geocode that is use to find the latitude of the input city.

Long=[loc1.longitude,loc.longitude] #loc1 or loc is object of geocode that is use to find the longitude of the input city.

gmap1(lat,Long) #calling gmap1 function.

n=input("if route is confrmed then type Y or y else type anything ")

if n.lower()=="y": #check the route is confermed or not.

a=geodesic((loc1.latitude,loc1.longitude), (loc.latitude,loc.longitude)) #geodesic is class of geopy that is use to calvulate distance between both city.

b=[]

b.append(str(a))

z=b[0].split('.')

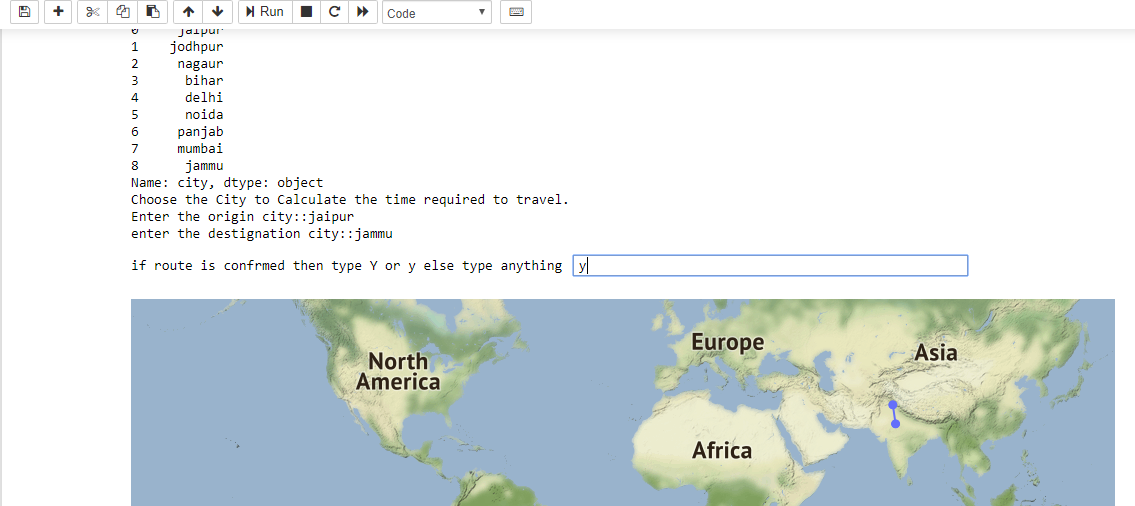
#print(z[0])

speed(z[0])

else:

main()

**RQ1-Output:**

****

**RQ2-Code:-**

def speed(a):

l=[]

a=int(a)

m=[]

#check all the transportation method on the basis of km/min.

hl=a/18 #for Hyperloop one

h2=hl//60

h21=hl%60

l.append([int(h2),round(h21)])

hs=a/5 #for HSR

hs2=hs//60

hs21=hs%60

l.append([int(hs2),round(hs21)])

rs=a/3.667 #For rails.

r2=rs//60

r21=rs%60

l.append([int(r2),round(r21)])

fs=a/14.0167 #for flight.

f2=fs//60

f21=fs%60

l.append([int(f2),round(f21)])

cs=a/1.867 #for car.

c2=a//60

c21=a%60

l.append([int(c2),round(c21)])

m=[hl,hs,rs,fs,cs]#Store all the timerequired to travel of different transportation in list.

s=[' Virgin Hyper loop one',' HSR',' Rail',' Air',' Car']

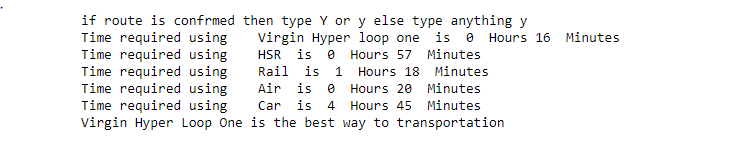
for i in range(len(m)):

print("Time required using ",s[i]," is ",l[i][0]," Hours",l[i][1]," Minutes")

print("Virgin Hyper Loop One is the best way to transportation")

tur\_layout(m,l) #call tur\_layout function.

**RQ2-output**

****

**RQ3-Code**

m=[hl,hs,rs,fs,cs]#Store all the timerequired to travel of different transportation in list.

s=[' Virgin Hyper loop one',' HSR',' Rail',' Air',' Car']

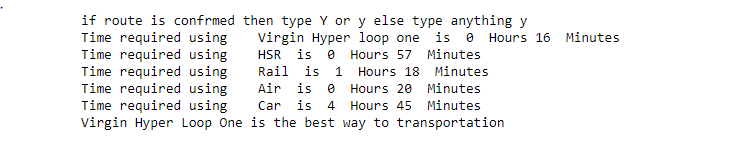
for i in range(len(m)):

print("Time required using ",s[i]," is ",l[i][0]," Hours",l[i][1]," Minutes")

print("Virgin Hyper Loop One is the best way to transportation")

tur\_layout(m,l)

**RQ3-Output**



**RQ7-Code:**

def main():

df=pd.read\_csv("city1.csv") #Read city1.csv file data from Pandas class.

print(df['city'][0:9])

print("Choose the City to Calculate the time required to travel.")

geolocator = Nominatim(timeout=4)

city1=input('Enter the origin city::') # take Source city name.

city =input('enter the destignation city::') #Take destignation city name.

if city1.lower()=='na' or city.lower()=='na':#Check the city name is vaiid or not.

print('invalid input city name')

exit

else:

loc1 = geolocator.geocode(city1) # Geocode class is use to calculate the cordinates of the city.

loc = geolocator.geocode(city)

lat=[loc1.latitude,loc.latitude] #loc1 or loc is object of geocode that is use to find the latitude of the input city.

Long=[loc1.longitude,loc.longitude] #loc1 or loc is object of geocode that is use to find the longitude of the input city.

gmap1(lat,Long) #calling gmap1 function.

n=input("if route is confrmed then type Y or y else type anything ")

if n.lower()=="y": #check the route is confermed or not.

a=geodesic((loc1.latitude,loc1.longitude), (loc.latitude,loc.longitude)) #geodesic is class of geopy that is use to calvulate distance between both city.

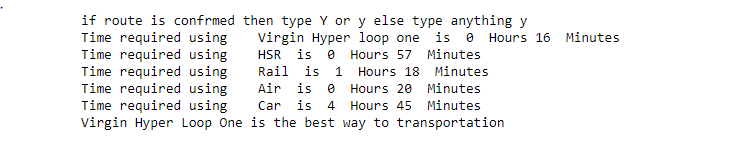
b=[]

b.append(str(a))

z=b[0].split('.')

#print(z[0])

**RQ7-Output**

****

**RQ8-Code:**

def tur\_layout(m,l):

maxheight = max(m)

numbars = len(m)

border = 10

wn = turtle.Screen() #make wn object of screen class.

wn.setworldcoordinates(0-border, 0-border, 40\*numbars+border, maxheight+border)

wn.bgcolor("white") #select background color

tess = turtle.Turtle() #make turtle class object.

tess.color("brown") #select border color.

tess.fillcolor("lightgreen") #select fill color

tess.pensize(1.3) #select size of pen or border.

s=[' Virgin Hyper loop one',' HSR',' Rail',' Air',' Car']

#create graph using matplotlib.

y\_pos = np.arange(len(s))

plt.barh(y\_pos, m)

plt.yticks(y\_pos, s)

plt.show()

print(" Horinzontal graph plot using matplotlib ")

#Create horizontal graph using pandas liabrary

df = pd.DataFrame({"sources":s, 'Time(in minutes)':m})

ax = df.plot.barh(x='sources', y='Time(in minutes)', rot=0)

print("Horizontal plot using pandas lib")

#create horizontal bar grapf using turtle.

for a in range(len(s)-1,-1,-1):

b="".join([str(l[a][0])," hour ",str(l[a][1])," minutes "])

drawBar(tess,m[a],b,s[a]) #Call the drawBar Function.

wn.exitonclick()

#drawBar function use for creating Turtle Graph.

def drawBar(t,height,b,n):

t.begin\_fill()

t.forward(height)

t.left(90)

t.write(" " + str(b))

t.forward(20)

t.write(str(n))

t.forward(4)

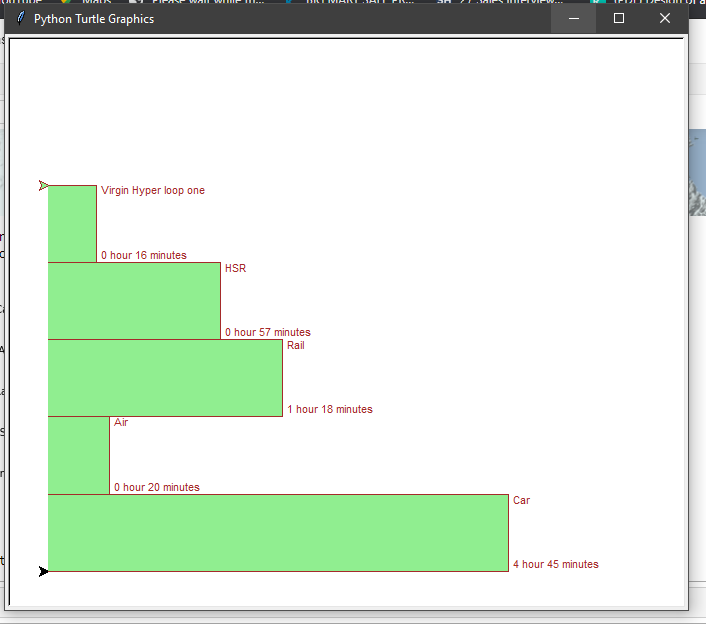
t.left(90)

t.forward(height)

t.right(180)

t.end\_fill()

**RQ-8-Output**

****

**RQ9-Code:**

s=[' Virgin Hyper loop one',' HSR',' Rail',' Air',' Car']

#create graph using matplotlib.

y\_pos = np.arange(len(s))

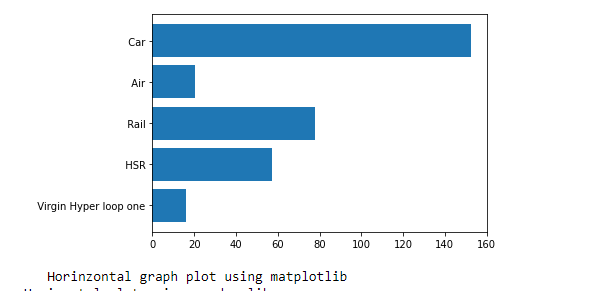
plt.barh(y\_pos, m)

plt.yticks(y\_pos, s)

plt.show()

print(" Horinzontal graph plot using matplotlib ")

**RQ-9-Output**

****

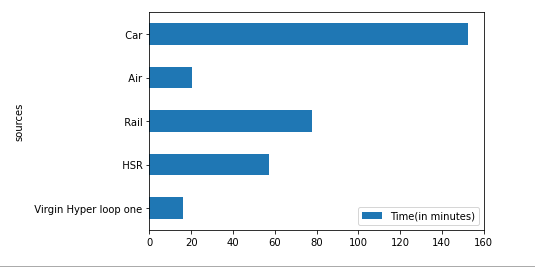
**RQ-10-Code**

#Create horizontal graph using pandas liabrary

df = pd.DataFrame({"sources":s, 'Time(in minutes)':m})

ax = df.plot.barh(x='sources', y='Time(in minutes)', rot=0)

print("Horizontal plot using pandas lib")

**RQ-10-Output**

**References:**

# A simulation-based approach to provide insights on Hyperloop network operations. About us. [Online]. Available at: <https://www.sciencedirect.com/science/article/pii/S2590198220300038>

# Route-estimator. About us. [Online]. Available at: <https://hyperloop-one.com/route-estimator/>

# [Suchithra Rajendran](https://www.sciencedirect.com/science/article/pii/S2590198220300038" \l "!) (2019). [Transportation Research Interdisciplinary Perspectives](https://www.sciencedirect.com/science/journal/25901982): Hyperloop network operations

# Plotly maps with Matplotlib Basemap in Python/v3.About us. [Online].Available at : <https://plotly.com/python/v3/ipython-notebooks/basemap-maps/>