TUGAS KECIL 3
IF2211 STRATEGI ALGORITMA

Implementasi Algoritma A* untuk Menentukan Lintasan Terpendek

Semester II
Tahun Ajaran 2020/2021

Disusun oleh :

Ridho Daffasyah	13519038
Thomas Ferdinand Martin	13519099

TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

2021

A. Kode Program

```
# Ridho Daffasyah 13519038
# Thomas Ferdinand Martin 13519099
# Algoritma Pencarian Jalur A*

import networkx as nx
import matplotlib.pyplot as plt
import math
from collections import deque
%matplotlib notebook

'''INISIASI CLASS'''
class Node:

    def __init__(self, name):
        self.name = name
        self.neighbours = []
        self.f = 0
        self.g = 0
        self.h = 0
        self.x = 0
        self.y = 0
    def AddNeighbour(self, neighbour):
        self.neighbours.append(neighbour)
    def GetG(self, g):
        return self.g
    def GetH(self, h):
        return self.h
    def GetNeighbour(self):
        return self.neighbours
    def FindF(self):
        return self.g + self.h

class Graph:

    def __init__(self):
        self.ListOfNode = []
        self.ListOFNodePosition = []
        self.AdjMatrix = []
        self.nNode = 0
    def InsertNode(self, node):
```

```
        self.ListOfNode.append(node)
    def GetNode(self, idx):
        try:
            return self.ListOfNode[idx]
        except:
            return None
    def GetNodeNeighbour(self, idx):
        return self.ListOfNode[idx].GetNeighbour()
    def ReadFromFile(self, filename):
        try:
            f = open("" + filename, "r")
        except:
            print("Salah memasukkan nama file")
            return
        i = 0
        lines = f.readlines()
        # Get n node
        self.nNode = int(lines[0].replace("\n", ""))
        # Get position
        for i in range (self.nNode):
            line = lines[i+1].replace("\n", "").split("|")
            # ["A", "2,3"]
            # versi 1
            node = Node(line[0])
            node.x = line[1].split(",")[0]
            node.y = line[1].split(",")[1]
            self.ListOfNode.append(node)
            # versi 2
            x = line[1].split(",")[0]
            y = line[1].split(",")[1]
            self.ListOFNodePosition.append((float(x), float(y)))
        # Get tetangga
        i = 0
        for i in range (self.nNode):
            line = lines[self.nNode+1+i].replace("\n", "").split(",")
            # Get Node
            node = self.ListOfNode[i]
            j = 0
            for adj in line[1:]:
                if adj == "1":
                    node.AddNeighbour(self.ListOfNode[j])
                j+=1
            self.ListOfNode[i] = node
```

```
    def PrintGraph(self):
        for node in self.ListOfNode:
            print(node.name, end =" ")
            for neighbour in node.neighbours:
                print(neighbour.name, end=" ")
            print("")

    def distance(self, node1, node2):
        if node1 == None or node2 == None:
            return 0
        return math.sqrt(pow((float)(node1.x)-(float)(node2.x),2) +
pow((float)(node1.y)-(float)(node2.y),2))

    def HaversineDistance(self, node1, node2):
        R = 6378137
        dlat = self.rad((float)(node1.x) - (float)(node2.x))
        dlong = self.rad((float)(node1.y) - (float)(node2.y))
        a = math.sin(dlat/2) * math.sin(dlat/2) +
math.cos(self.rad(node1.x))*math.cos(self.rad(node2.x))*math.sin(dlong/2)*m
ath.sin(dlong/2)
        c = math.asin(math.sqrt(a))
        d = R *c
        return d

    def rad(self, x):
        return (float)(x)*math.pi/180

    def bobot(self, node_name1, node_name2):
        node1 = self.ListOfNode[self.GetNodeIdx(node_name1)]
        node2 = self.ListOfNode[self.GetNodeIdx(node_name2)]
        if node1.name in node2.neighbours:
            return self.distance(node1, node2)
        return 0

    def IsNodeInSet(self, node, list_node):
        for nodes in list_node:
            if node == nodes:
                return True
        return False

    def GetNodeIdx(self, node_name):
        i = 0
        for node in self.ListOfNode:
```

```
            if node.name == node_name:
                return i
            i+=1

    def GetListOfNode(self):
        return self.ListOfNode

    def getLowestF(self, openSet, fScore):
        lowest = float("inf")
        lowestNode = None
        for node in openSet:
            if fScore[node] < lowest:
                lowest = fScore[node]
                lowestNode = node
        return lowestNode

    def reconstructPath(self, cameFrom, goal):
        path = deque()
        node = goal
        path.appendleft(node)
        while node in cameFrom:
            node = cameFrom[node]
            path.appendleft(node)
        return path
    def getNodeFromName(self, node_name):
        for node in self.ListOfNode:
            if node.name == node_name:
                return node

def aStar(graph, start, goal):
    #Inisialisasi open list dan closed list
    start = graph.getNodeFromName(start)
    goal = graph.getNodeFromName(goal)

    cameFrom = {}
    openSet = set([start])
    closedSet = set()

    gScore = {} #untuk simpan nilai g
    fScore = {} #untuk simpan nilai f

    gScore[start] = 0 #set g score start node dengan 0
    fScore[start] = gScore[start]+graph.HaversineDistance(start,goal) #set
```

```
f score start node

    #iterasi selama openset tidak 0
    while (len(openSet) != 0):
        #set current node dengan node yang punya lowest f
        current = graph.getLowestF(openSet, fScore)
        #jika current node sama dengan yang dicari
        if current == goal:
            return (graph.reconstructPath(cameFrom, goal), fScore[current])
        #remove current node dari openSet
        openSet.remove(current)
        #tambahkan current node ke closedSet
        closedSet.add(current)
        #lakukan pencarian nilai f ke semua tetangga dari current node
        if (len(goal.neighbours)!=0):
            for neighbour in current.neighbours:
                node_neighbour = graph.getNodeFromName(neighbour.name)
                tentative_gScore = gScore[current] +
graph.HaversineDistance(current, node_neighbour)
                if node_neighbour in closedSet:
                    continue
                if node_neighbour in gScore:
                    if tentative_gScore > gScore[node_neighbour]:
                        continue
                if node_neighbour not in closedSet or tentative_gScore <
gScore[node_neighbour]:
                    cameFrom[node_neighbour] = current
                    gScore[node_neighbour] = tentative_gScore
                    fScore[node_neighbour] = gScore[node_neighbour] +
graph.HaversineDistance(node_neighbour,goal)
                    if node_neighbour not in openSet:
                        openSet.add(node_neighbour)
        else :
            print("Tidak bisa akses kesana")
    path = deque()
    return (path, 0)

# Tuple node untuk visualisasi graf
def TupleOfEdge(listofnode, matrix, listofpos):
    listoftuple = []
    for i in range(len(listofnode)):
        for j in range(i, len(listofnode)):
            if matrix[i][j] == '1':
```

```
                edge = (listofnode[i], listofnode[j], {'weight':
round(HaversineDistance(i, j, listofpos), 2), 'node_size': 500})
                listoftuple.append(edge)
    return listoftuple

def rad(x):
    return x*math.pi/180
# Tidak terpakai
def LongLatToXY(long, lat):
    return (6378137*math.cos(lat)*math.cos(long),
6378137*math.cos(lat)*math.sin(long))
def LongLatToX(long, lat):
    return (6378137*math.cos(lat)*math.cos(long))
def LongLatToY(long, lat):
    return (6378137*math.cos(lat)*math.sin(long))
def PrintHeuristic(nodes, nodespos, goal_node):
    idx = 0
    for i in range(len(nodes)):
        if nodes[i] == goal_node:
            idx = i
            break
    print("Jarak dari " + goal_node + " dalam meter : ")
    i = 0
    for node in nodespos:
        print(nodes[i], end=" : ")
        print(HaversineDistance(idx, i, nodespos))
        i+=1

'''INPUT NAMA FILE'''
filepath = "../test/"
file = input("Masukkan nama file: ")

''' FOR VISUALIZATION '''
astar_graph = Graph()
filename = filepath + file
try:
    f = open(filename, "r")
    astar_graph.ReadFromFile(filename)
    i = 0
    lines = f.readlines()
    ListOfNode = []
    ListOfNodePosition = []
    # Get n node
```

```
    nNode = int(lines[0].replace("\n", ""))
    AdjMatrix = [[] for i in range (nNode)]
    # Get position
    for i in range (nNode):
        line = lines[i+1].replace("\n", "").split("|")
        x = line[1].split(",")[0]
        y = line[1].split(",")[1]
        ListOfNodePosition.append((float(x), float(y)))
    # Get tetangga
    for i in range (nNode):
        line = lines[nNode+1+i].replace("\n", "").split(",")
        adj = line[1::]
        # Get Node
        ListOfNode.append(line[0])
        AdjMatrix[i] = adj
except:
    print("Salah memasukkan nama file")

''' GRAPH VISUALIZATION '''
%matplotlib notebook
plt.figure(figsize=(8, 8))
G = nx.Graph()

# Menambahkan node
idx = 0
for node in ListOfNode:
    lat = ListOfNodePosition[idx][0]
    long = ListOfNodePosition[idx][1]
    G.add_node(node, pos=(lat*100, long*100))
    idx+=1

# Menambahkan sisi pada graf dengan bobot masing-masing sisi
G.add_edges_from(TupleOfEdge(ListOfNode, AdjMatrix, ListOfNodePosition))

# label untuk bobot pada graf
labels = nx.get_edge_attributes(G, "weight")

# Layout untuk menggambarkan graf dengan representasi pos
position=nx.get_node_attributes(G,'pos')
layout = nx.spring_layout(G, pos=position)
nx.draw(G, layout, with_labels=True, font_size = 7)

# Menambahkan label bobot
```

```
nx.draw_networkx_edge_labels(G, pos=layout, edge_labels=labels, font_size =
6)

plt.axis("off")
plt.title("Tampilan Peta")
plt.show()

''' INPUT NODE ASAL DAN TUJUAN '''
print("Daftar simpul: ")
index = 1
for node in ListOfNode:
    print(index, end="")
    print(". " + node)
    index+=1
start_node = input("Masukkan nama node asal: ")
goal_node = input("Masukkan nama node tujuan: ")
print()
# Perhitungan path
try:
    (path,weight) = aStar(astar_graph, start_node, goal_node)
except:
    print("Titik awal atau tujuan tidak sesuai")
if (path):
    print("Jalur : ")
    for i in range(len(path)):
        print(path[i].name)
    print("Total Jarak : ", end="")
    print(weight, end=" ")
    print("meter")
else:
    print("Tidak ada jalur")


print()
''' ROUTE VISUALIZATION '''
%matplotlib notebook

# Tampilkan h(n)
PrintHeuristic(ListOfNode, ListOfNodePosition, goal_node)

routes = []
for route in path:
    routes.append(route.name)
```

```
# make a tuple untuk visualisasi
edges = []
for i in range (len(routes)-1):
    edges.append((routes[i], routes[i+1]))

plt.figure(figsize=(8, 8))

# sisi graf
nx.draw_networkx_edges(
    G, layout, edgelist=edges, width=6, alpha=0.5, edge_color="b",
style="dashed"
)

# label sisi
nx.draw_networkx_edge_labels(G, pos=layout, edge_labels=labels, font_size =
8)

# warna node
color_map = []
for node in G:
    if node == start_node or node == goal_node:
        color_map.append('red')
    else:
        if node in routes:
            color_map.append('blue')
        else:
            color_map.append('green')
# insert layout
nx.draw(G, layout, node_color=color_map, with_labels=True, font_size = 6)
#plt.xlim(-1.5, 1.5)
#plt.ylim(-1.5, 1.5)
plt.axis("off")
plt.show()

"""DRAW MAP"""
from ipyleaflet import (
    Map, basemaps, basemap_to_tiles,
    Circle, Marker, Rectangle, LayerGroup, AntPath
)
from ipywidgets import HTML

arraypos = []
```

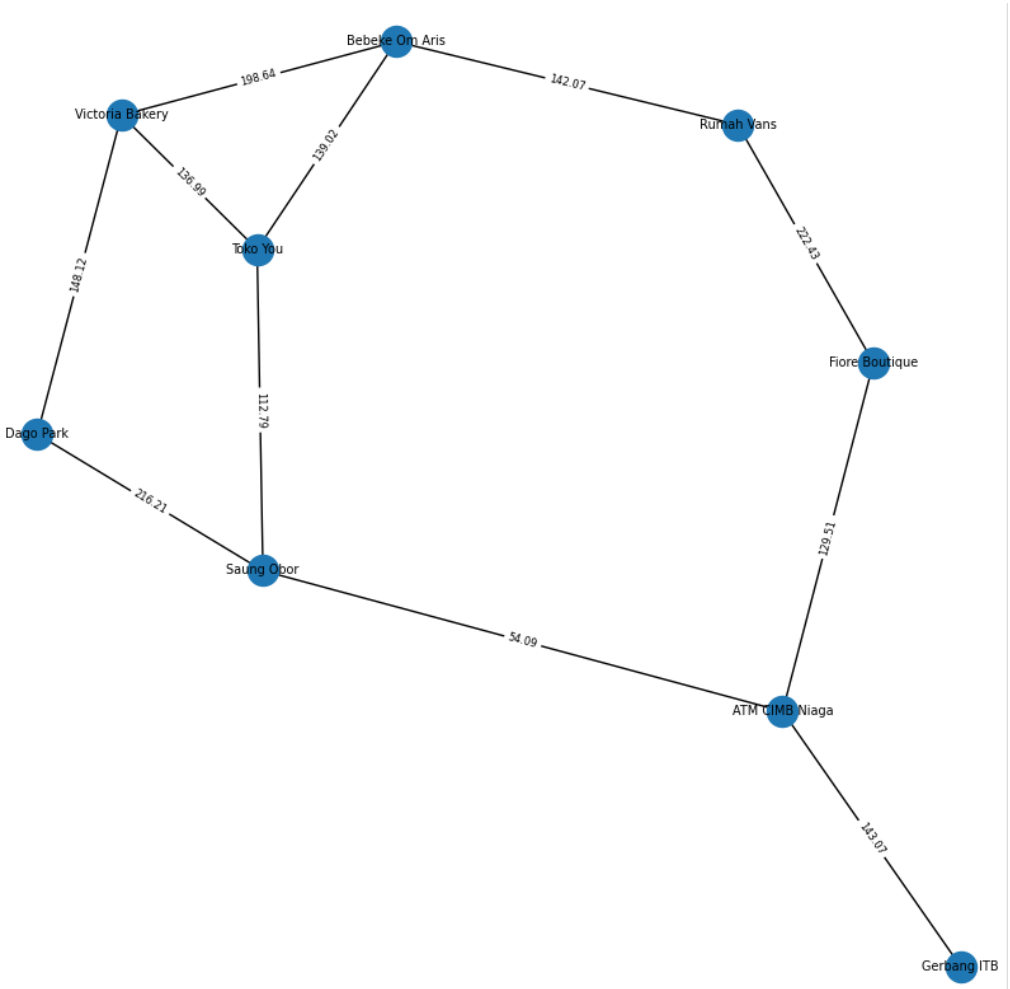
```
for i in range(len(astar_graph.ListOfNode)):
    coord = [float(ListOfNodePosition[i][0]),
float(ListOfNodePosition[i][0])]
    arraypos.append(coord)

# map center
center = ListOfNodePosition[0]
m = Map(center=center, zoom=15)

# map marker min 2
# TODO Label?, icon
marker = Marker(location=center, draggable=False)
marke2 = Marker(location=ListOfNodePosition[1], draggable=False)

# Create layer group
layer_group = LayerGroup(layers=(marker, marke2))
markers = []

# Create some layers
for position in ListOfNodePosition:
    marker = Marker(location=position, draggable=False)
    markers.append(marker)
    layer_group.add_layer(marker)

labels = []
if (path):
    for line in ListOfNode:
        message = HTML()
        labels.append(message)
        message.value = line
    i = 0
    for mark in markers:
        mark.popup = labels[i]
        i+=1
m.add_layer(layer_group)
m


"""DRAW ROUTE"""
# Get position based on path
coord = []
if(path):
    for i in range(len(path)):
```

```
        # catat posisi node
        point = [float(path[i].x), float(path[i].y)]
        coord.append(point)


# Draw Path
mPath = Map(center=(path[0].x, path[0].y), zoom=15)
markerStart = Marker(location=(path[0].x, path[0].y), draggable=False,
rise_on_hover=True)
markerGoal = Marker(location=(path[len(path)-1].x, path[len(path)-1].y),
draggable=False, rise_on_hover=True)

layer_group = LayerGroup(layers=(markerStart, markerGoal))
mPath.add_layer(layer_group)

#TODO: Hover pop up label
messageStart = HTML()
messageGoal = HTML()

messageStart.value = path[0].name
messageGoal.value = path[len(path) - 1].name

markerStart.popup = messageStart
markerGoal.popup = messageGoal

ant_path = AntPath(
    locations=coord,
    dash_array=[1, 10],
    delay=1000,
    color='#7590ba',
    pulse_color='#3f6fba'
)
layer_group.add_layer(ant_path)

mPath
```

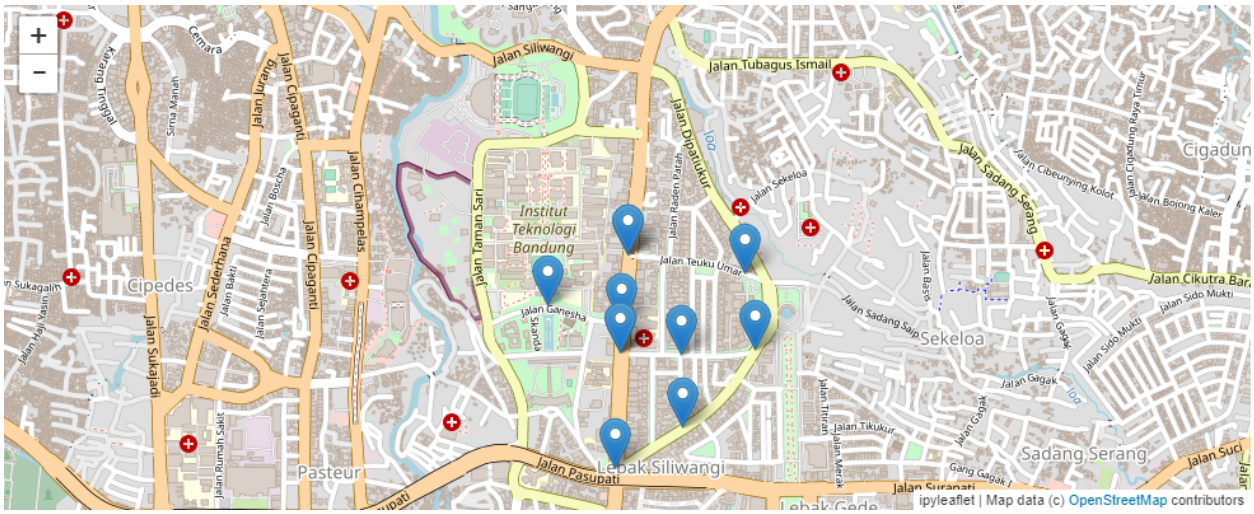
B. Graf Input

1. Peta ITB

a. Graf berbobot peta jalan sekitar ITB:



b. Tampilan pada map:



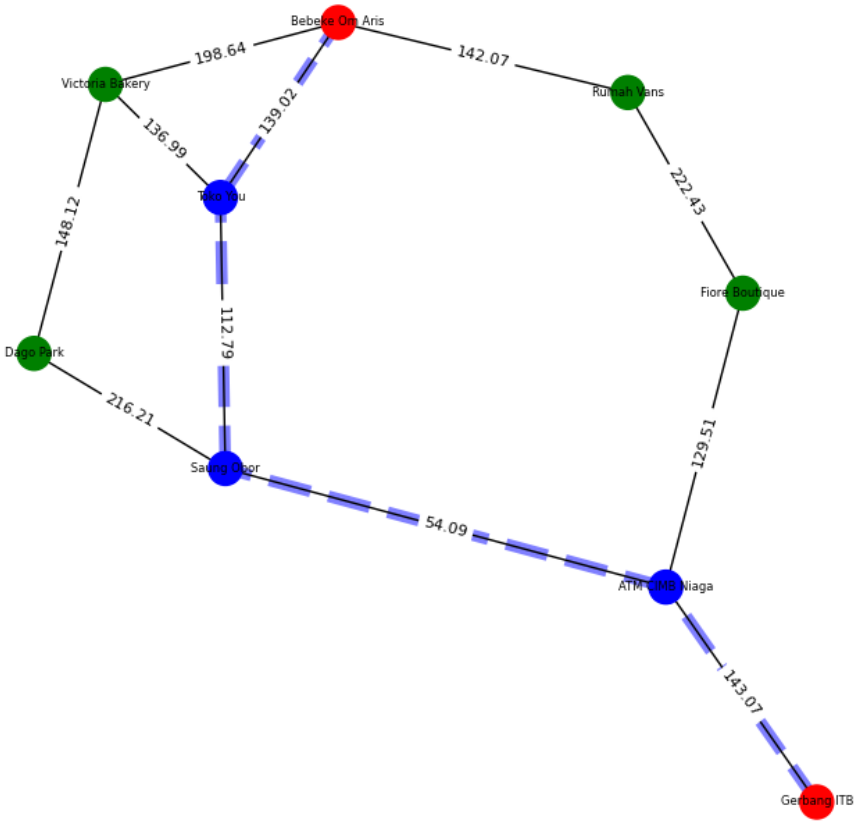
c. File input:

```
9
Fiore Boutique|-6.89148,107.61313
Saung Obor|-6.89477,107.61289
Toko You|-6.89484,107.61493
Dago Park|-6.89865,107.6127
Victoria Bakery|-6.8973,107.61501
Bebeke Om Aris|-6.89467,107.61744
ATM CIMB Niaga|-6.8938,107.61295
Gerbang ITB|-6.89321,107.61043
Rumah Vans|-6.89214,107.6171
Fiore Boutique,0,0,0,0,0,0,1,0,1
Saung Obor,0,0,1,1,0,0,1,0,0
Toko You,0,1,0,0,1,1,0,0,0
Dago Park,0,1,0,0,1,0,0,0,0
Victoria Bakery,0,0,1,1,0,1,0,0,0
Bebeke Om Aris,0,0,1,0,1,0,0,0,1
ATM CIMB Niaga,1,1,0,0,0,0,0,1,0
Gerbang ITB,0,0,0,0,0,0,1,0,0
Rumah Vans,1,0,0,0,0,1,0,0,0
```

d. Input dari simpul Gerbang ITB ke Bebeke Om Aris:

```
Daftar simpul:
1. Fiore Boutique
2. Saung Obor
3. Toko You
4. Dago Park
5. Victoria Bakery
6. Bebeke Om Aris
7. ATM CIMB Niaga
8. Gerbang ITB
9. Rumah Vans
Masukkan nama node asal: Gerbang ITB
Masukkan nama node tujuan: Bebeke Om Aris

Jalur :
Gerbang ITB
ATM CIMB Niaga
Saung Obor
Toko You
Bebeke Om Aris
Total Jarak : 448.9702270008344 meter

Jarak dari Bebeke Om Aris dalam meter :
Fiore Boutique : 297.06158867993724
Saung Obor : 251.48203029533326
Toko You : 139.01804533770667
Dago Park : 343.03765604291306
Victoria Bakery : 198.6411556782538
Bebeke Om Aris : 0.0
ATM CIMB Niaga : 252.7866600883181
Gerbang ITB : 395.78621925439415
Rumah Vans : 142.06690426539743
```

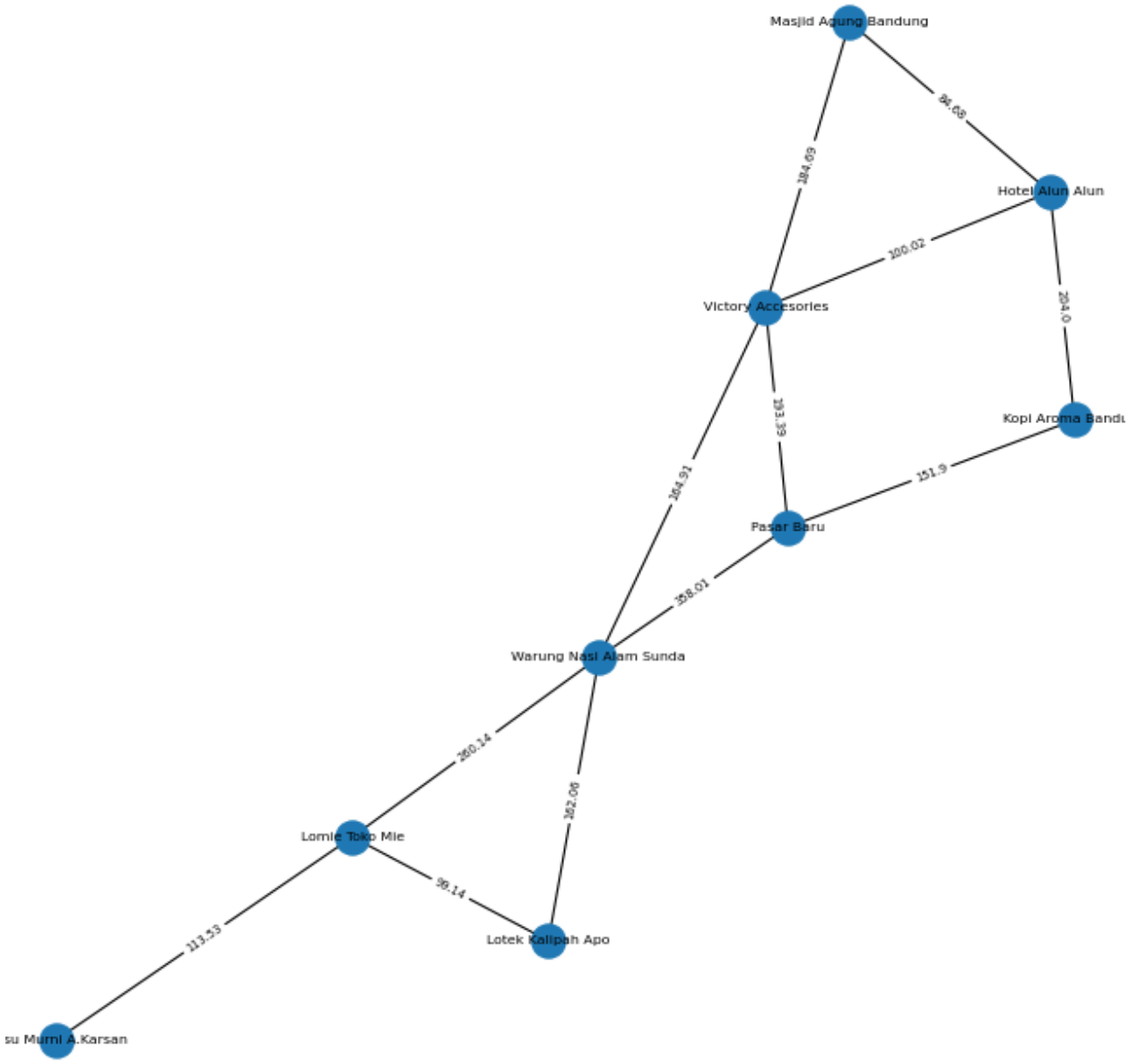
e. Visual Graf



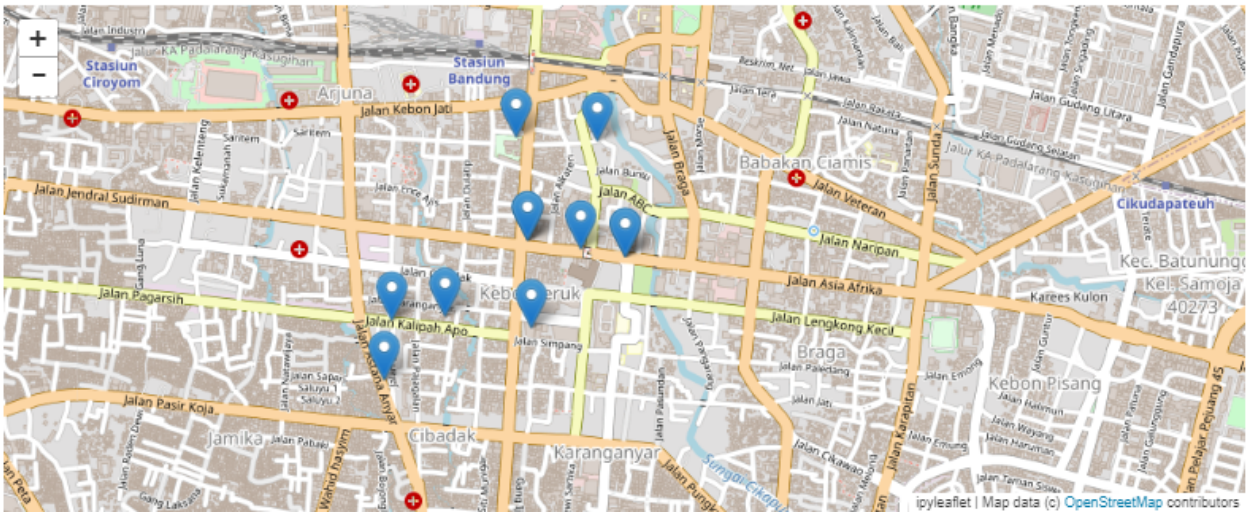
f. Tampilan pada Peta

2. Peta Alun-alun Bandung

a. Graf berbobot peta Alun-alun Bandung



b. Tampilan pada map



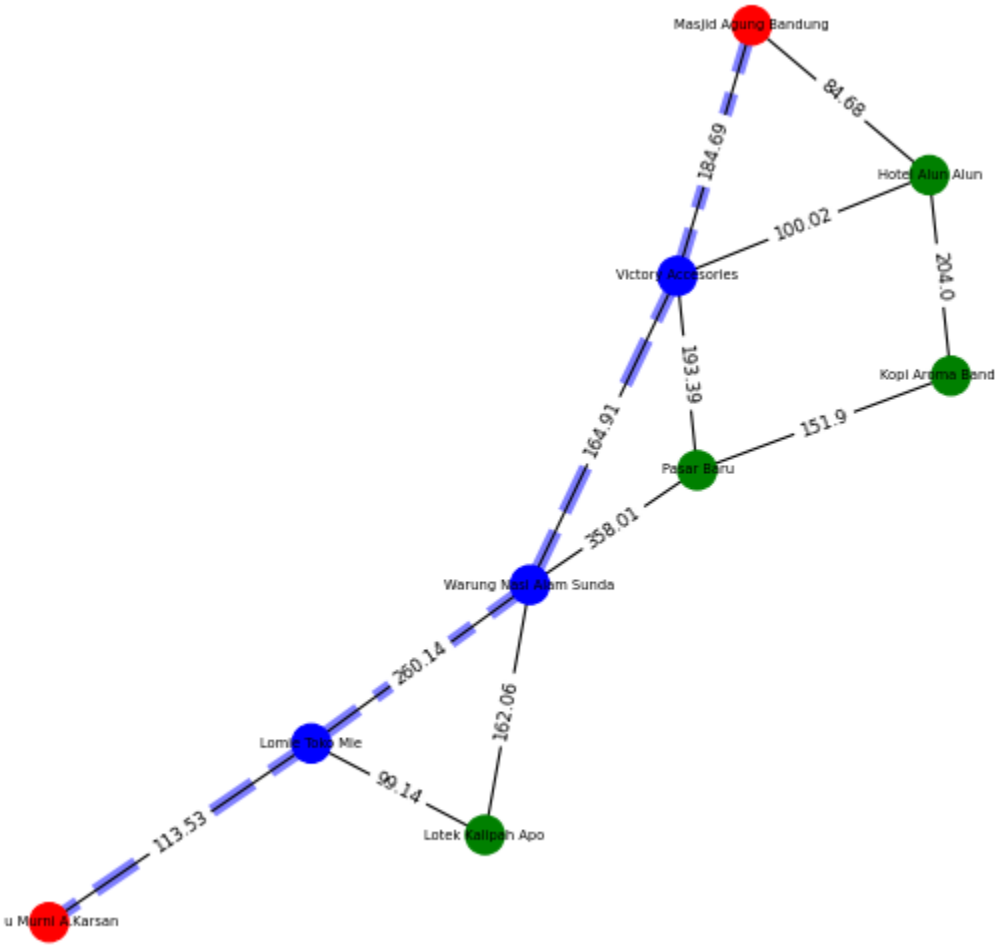
c. File Input

```
9
Masjid Agung Bandung|-6.921281496896509,107.60764937328989
Victory Accesories|-6.920717012051872,107.60435562083042
Warung Nasi Alam Sunda|-6.923677887307494,107.60446290919646
Lotek Kalipah Apo|-6.923294465659959,107.6015553948104
Lomie Toko Mie|-6.923390321141749,107.5997636791993
Susu Murni A.Karsan|-6.9254139314640195,107.59950618715034
Pasar Baru|-6.917266184334281,107.60394792505646
Kopi Aroma Bandung|-6.917383342476477,107.60669450695525
Hotel Alun Alun|-6.92100724252596,107.60614197196787
Masjid Agung Bandung,0,1,0,0,0,0,0,0,1
Victory Accesories,1,0,1,0,0,0,1,0,1
Warung Nasi Alam Sunda,0,1,0,1,1,0,1,0,0
Lotek Kalipah Apo,0,0,1,0,1,0,0,0,0
Lomie Toko Mie,0,0,1,1,0,1,0,0,0
Susu Murni A.Karsan,0,0,0,0,1,0,0,0,0
Pasar Baru,0,1,1,0,0,0,0,1,0
Kopi Aroma Bandung,0,0,0,0,0,0,1,0,1
Hotel Alun Alun,1,1,0,0,0,0,0,1,0
```

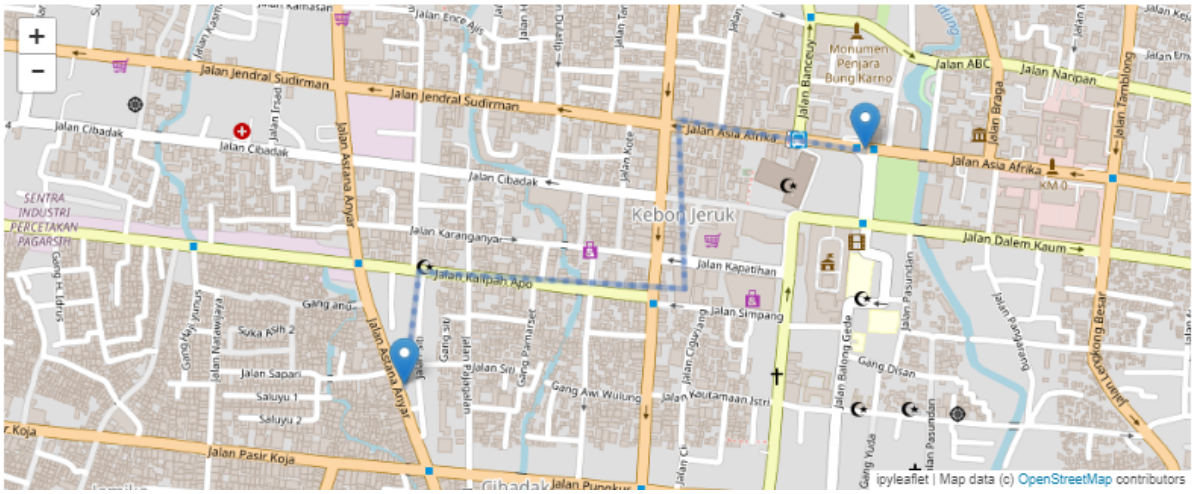
d. Input simpul dari Masjid Agung Bandung ke Susu Murni A.Karsan

```
Masukkan node asal: Masjid Agung Bandung
Masukkan node tujuan: Susu Murni A.Karsan

Jalur :
Masjid Agung Bandung
Victory Accesories
Warung Nasi Alam Sunda
Lomie Toko Mie
Susu Murni A.Karsan
Total Jarak : 723.266011959406 meter

Jarak dari Susu Murni A.Karsan dalam meter :
Masjid Agung Bandung : 505.3248182164872
Victory Accesories : 374.3562895662381
Warung Nasi Alam Sunda : 290.42349189475897
Lotek Kalipah Apo : 163.514213787177
Lomie Toko Mie : 113.52865226405426
Susu Murni A.Karsan : 0.0
Pasar Baru : 515.6517711343055
Kopi Aroma Bandung : 597.952302418558
Hotel Alun Alun : 441.12851670420423
```

e. Visual graf

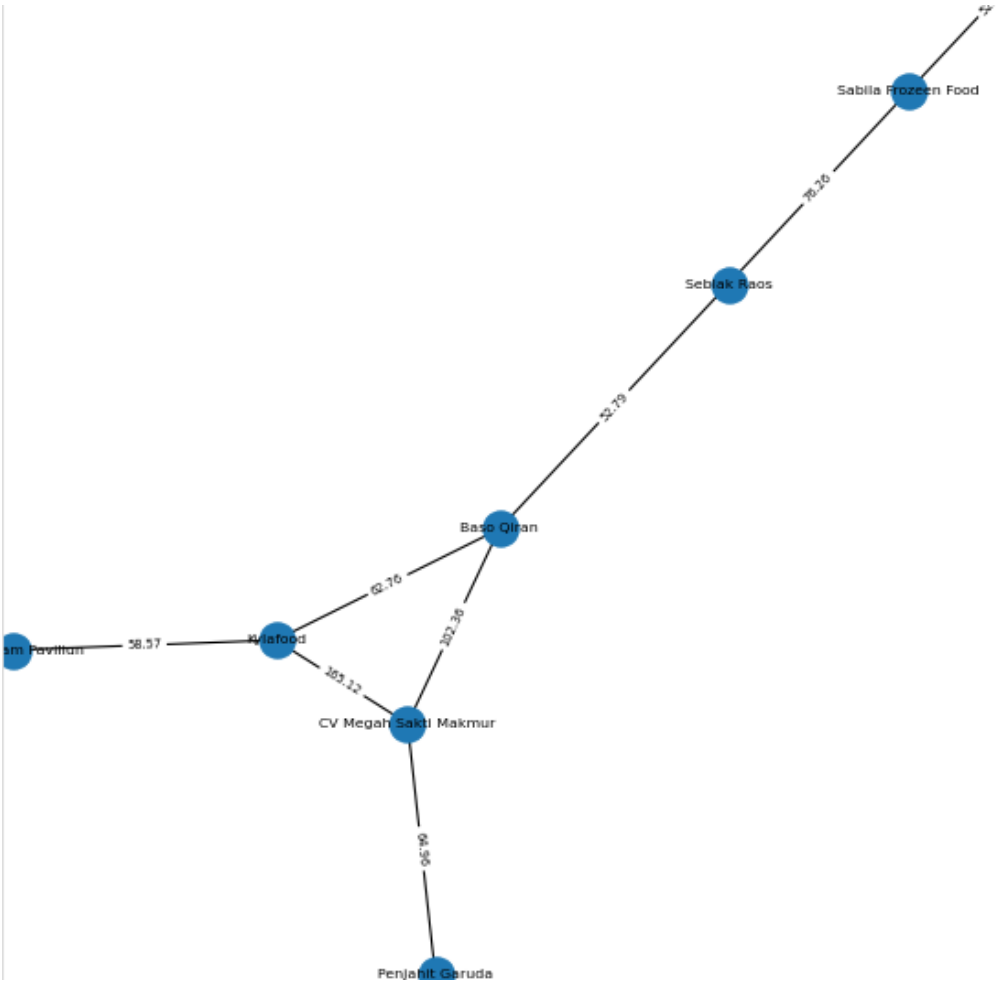


f. Tampilan pada peta

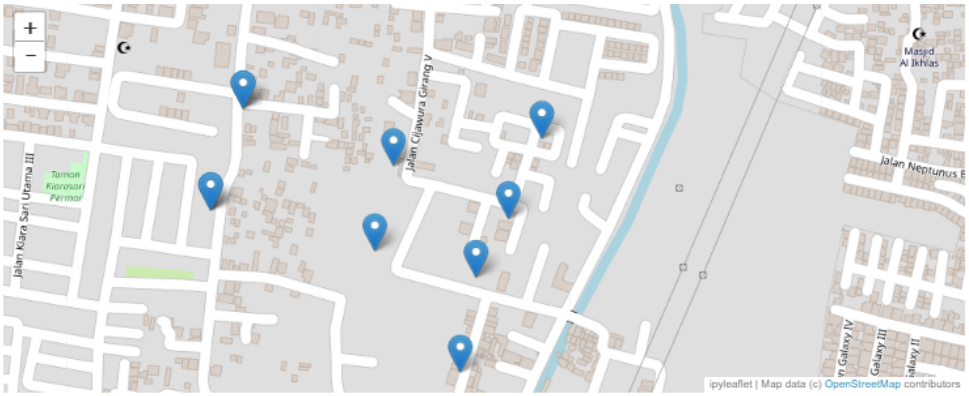


3. Peta Buahbatu

a. Graf berbobot peta Buahbatu



b. Tampilan pada map



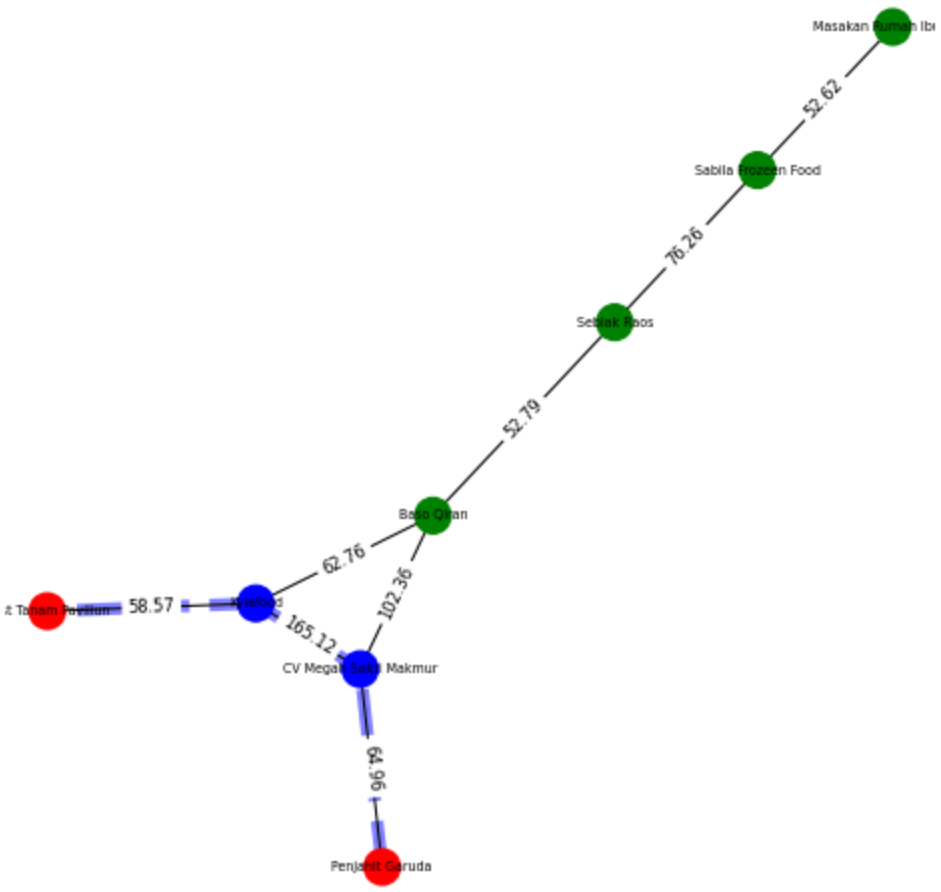
c. File Input

```
8
Penjahit Garuda|-6.947530837186421,107.64870581755453
CV Megah Sakti Makmur|-6.948643766902244,107.64835176602242
Baso Qiran|-6.949091068056918,107.65014884597933
Seblak Raos|-6.948164515182551,107.6503526938769
Sabila Frozeen Food|-6.948734292155658,107.65160796763203
Masakan Rumah Ibu Rien|-6.947860988847261,107.6519727480186
Kylafood|-6.9493732935721715,107.6512485517171
Kost Tanam Paviliun|-6.9504116689757955,107.65107689035113
Penjahit Garuda,0,1,0,0,0,0,0,0
CV Megah Sakti Makmur,1,0,1,0,0,0,1,0
Baso Qiran,0,1,0,1,0,0,1,0
Seblak Raos,0,0,1,0,1,0,0,0
Sabila Frozeen Food,0,0,0,1,0,1,0,0
Masakan Rumah Ibu Rien,0,0,0,0,1,0,0,0
Kylafood,0,1,1,0,0,0,0,1
Kost Tanam Paviliun,0,0,0,0,0,0,1,0
```

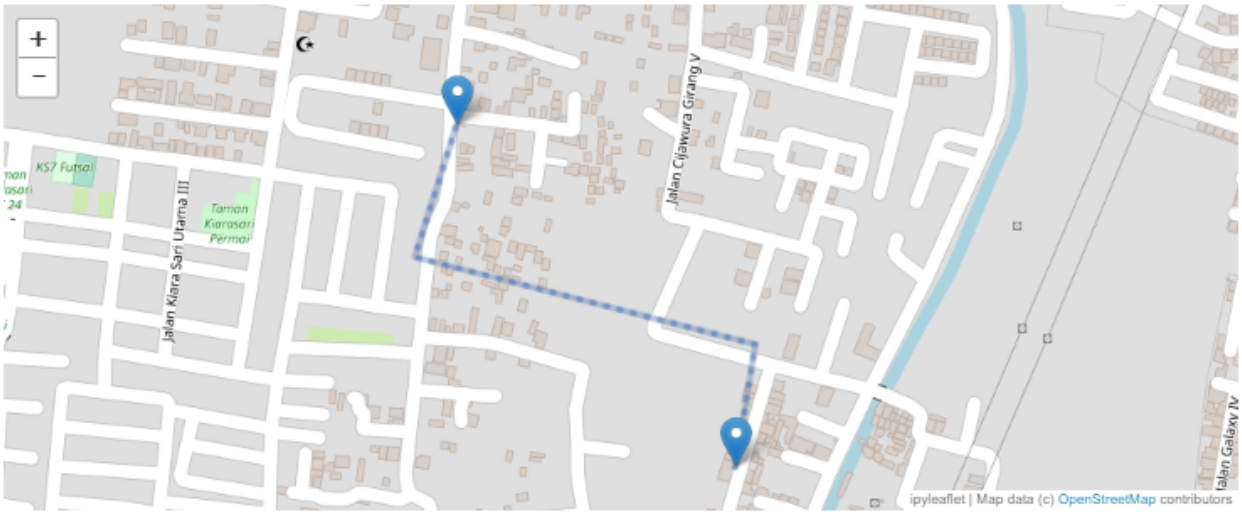
d. Input simpul dari Penjahit Garuda ke Kost Tanam Paviliun

```
Masukkan node asal: Penjahit Garuda
Masukkan node tujuan: Kost Tanam Paviliun

Jalur :
Penjahit Garuda
CV Megah Sakti Makmur
Kylafood
Kost Tanam Paviliun
Total Jarak : 288.6499313722112 meter

Jarak dari Kost Tanam Paviliun dalam meter :
Penjahit Garuda : 207.05789229862077
CV Megah Sakti Makmur : 179.86850340338302
Baso Qiran : 89.62158582796617
Seblak Raos : 131.32024021482366
Sabila Frozeen Food : 97.86475892619467
Masakan Rumah Ibu Rien : 150.35120166174616
Kylafood : 58.568749851312916
Kost Tanam Paviliun : 0.0
```

e. Visual Graf

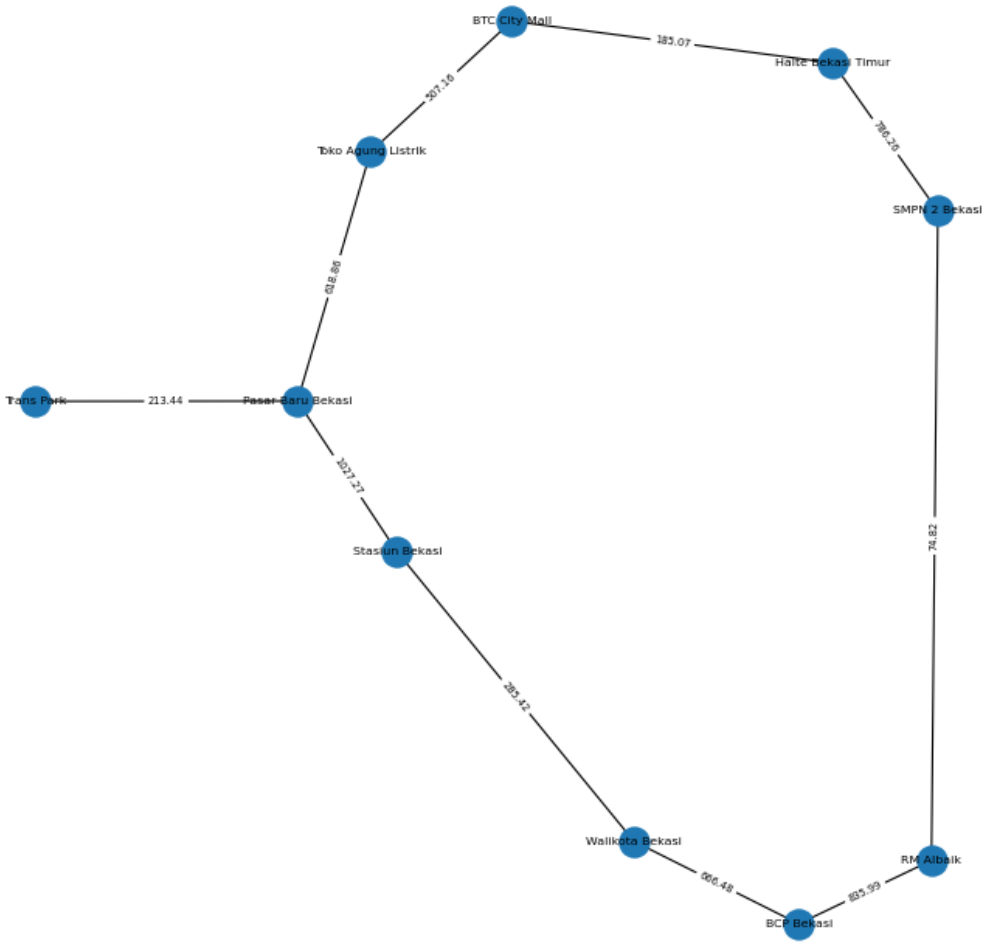


f. Tampilan pada peta

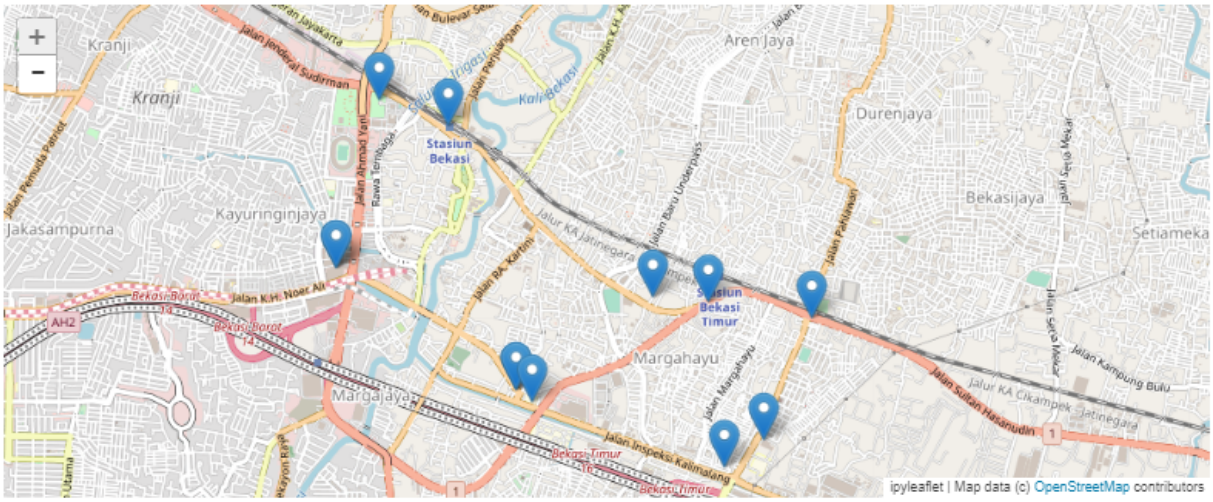


4. Peta Kota Bekasi

a. Graf berbobot peta Kota Bekasi :



b. Tampilan pada map :



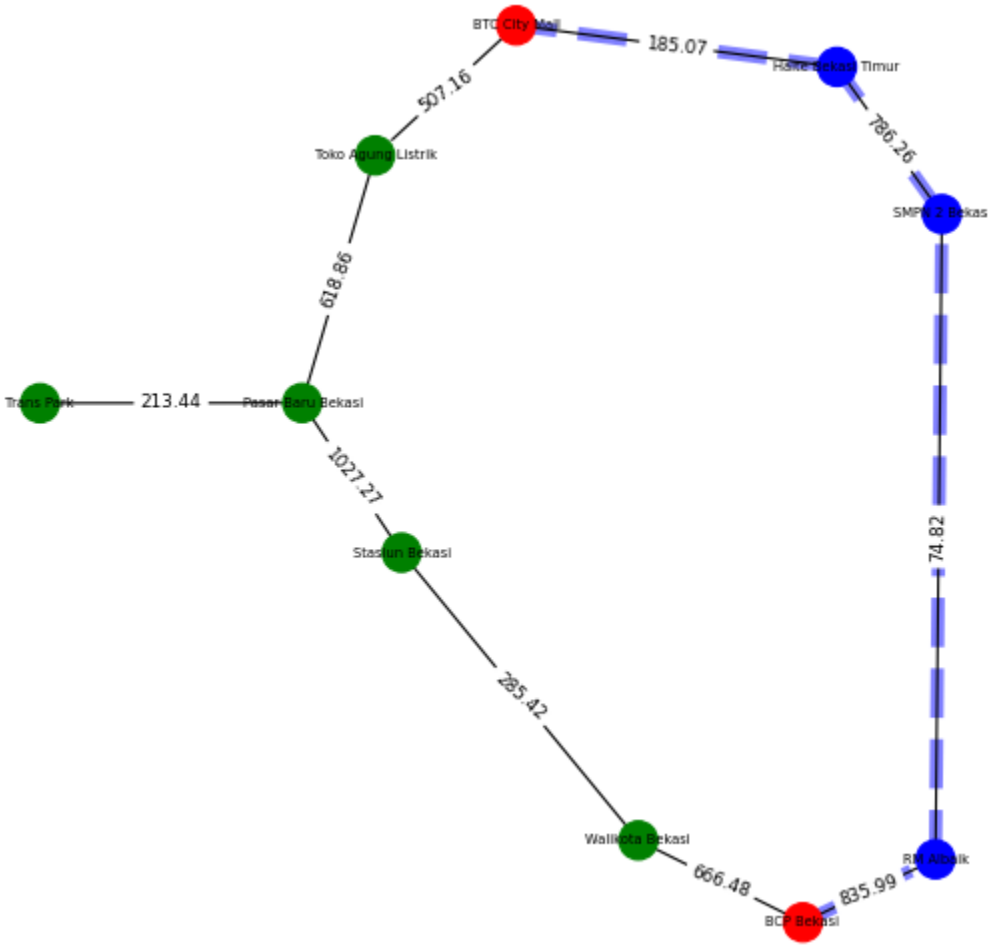
c. File Input :

```
10
BCP Bekasi|-6.246476216598358,106.99142746081678
RM Albaik|-6.254879360543884,107.00395091345622
SMPN 2 Bekasi|-6.255711227408035,107.00501306822787
Halte Bekasi Timur|-6.260309280448163,107.01845009642575
BTC City Mall|-6.258426929757609,107.02120740722451
Toko Agung Listrik|-6.249932721458352,107.02452473658256
Trans Park|-6.24880569171188,107.01727848591052
Pasar Baru Bekasi|-6.248495349743966,107.01343353641539
Stasiun Bekasi|-6.236641593254381,106.99920276703072
Walikota Bekasi|-6.234863385553174,106.9943643713209
BCP Bekasi,0,1,0,0,0,0,0,0,0,1
RM Albaik,1,0,1,0,0,0,0,0,0,0
SMPN 2 Bekasi,0,1,0,1,0,0,0,0,0,0
Halte Bekasi Timur,0,0,1,0,1,0,0,0,0,0
BTC City Mall,0,0,0,1,0,1,0,0,0,0
Toko Agung Listrik,0,0,0,0,0,1,0,1,0,0
Trans Park,0,0,0,0,0,1,0,1,0,0
Pasar Baru Bekasi,0,0,0,0,0,0,1,0,1,0
Stasiun Bekasi,0,0,0,0,0,0,0,1,0,1
Walikota Bekasi,1,0,0,0,0,0,0,0,1,0
```

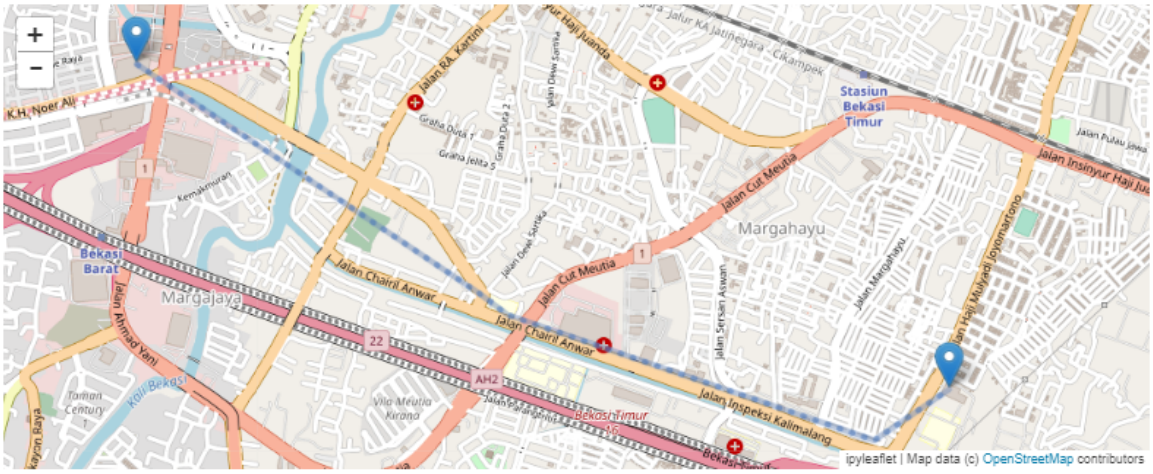
d. Input dari simpul BCP Bekasi ke BTC City Mall :

```
Masukkan node asal: BCP Bekasi
Masukkan node tujuan: BTC City Mall

Jalur :
BCP Bekasi
RM Albaik
SMPN 2 Bekasi
Halte Bekasi Timur
BTC City Mall
Total Jarak : 1882.1380539113961 meter

Jarak dari BTC City Mall dalam meter :
BCP Bekasi : 1776.8850880006623
RM Albaik : 974.975367072752
SMPN 2 Bekasi : 908.6636830662081
Halte Bekasi Timur : 185.06869356254134
BTC City Mall : 0.0
Toko Agung Listrik : 507.1627535023503
Trans Park : 577.9547820729655
Pasar Baru Bekasi : 700.4117310266762
Stasiun Bekasi : 1718.3190144108023
Walikota Bekasi : 1981.4074876875777
```

e. Visual graf :



f. Tampilan pada peta



C. Alamat Kode

https://github.com/thomas-fm/AStarMap.git

1	Program dapat menerima input graf	✔
2	Program dapat menghitung lintasan terpendek	✔
3	Program dapat menampilkan lintasan terpendek serta jaraknya	✔
4	Bonus: Program dapat menerima input peta dengan Google Map API dan menampilkan peta	(menampilkan saja)