Evaluating Public Transport Accessibility to Places of Interest in Melbourne

Authored by: Emmanuel Clement Anthony

Duration: {90} mins

Level: {Intermediate}

Pre-requisite Skills: {Python},{Geospatial Analysis},{Statistical Analysis},{Data Analysis}, {Machine Learning}

Scenario

As a: City Planner

I want: To assess and improve the accessibility of public transport (bus, tram, and train services) to key landmarks in Melbourne

So that: Residents and visitors can easily access important places of interest, leading to better connectivity, increased tourism, and enhanced quality of life.

What this use case will teach you

At the end of this use case you will:
- Cleaning and preprocessing data to ensure accuracy and consistency.
- Calculating distances between landmarks and various public transport stops.
- Developing an accessibility index to quantify the ease of access to public transport for each landmark.
- Visualizing data distributions and geographic patterns using histograms, scatter plots, and pair plots.
- Conducting correlation analysis to understand relationships between different transport modes.
- Applying clustering algorithms (e.g., KMeans) to group landmarks based on accessibility profiles.
- Identifying gaps in public transport coverage and providing actionable recommendations.
- Creating informative visualizations to communicate findings.
- Highlight the geographical distribution of tram and train stops and also shows the flow of pedestrians in the area.

Description

The City of Melbourne has numerous landmarks that attract residents and tourists alike. However, the current public transport services might not adequately serve all these key places of interest. By analyzing the proximity and accessibility of train stations, tram stops, and bus stops to these landmarks, we can identify areas where public transport coverage is insufficient. The goal is to provide actionable insights and recommendations for city planners and transport authorities to improve public transport connectivity. This will ensure that all significant landmarks are easily reachable, promoting a more integrated and efficient public transport network throughout the city.

Importing Libraries

In [43]:
```
import numpy as np
import pandas as pd
from io import StringIO
import requests
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
import folium
from folium.plugins import HeatMap
from IPython.display import IFrame
import plotly.graph_objs as go
import geopandas as gpd
import json
from shapely.geometry import Point
```

In [2]:
```
def API_Unlimited(datasetname, apikey):
    base_url =
'https://data.melbourne.vic.gov.au/api/explore/v2.1/catalog/datasets/'
    format = 'csv'

    url = f'{base_url}{datasetname}/exports/{format}'
    params = {
        'select': '*',
        'limit': -1,
        'lang': 'en',
        'timezone': 'UTC',
        'api_key': apikey
    }

    # GET request
    response = requests.get(url, params=params)

    if response.status_code == 200:
        # StringIO to read the CSV data
        url_content = response.content.decode('utf-8')
        df = pd.read_csv(StringIO(url_content), delimiter=';')
        print(df.sample(10, random_state=999))
        return df
    else:
        print(f'Request failed with status code {response.status_code}')
        return None

apikey = ''
datasets = {
    "landmarks": "landmarks-and-places-of-interest-including-schools-
theatres-health-services-spor",
    "bus_stops": "bus-stops",
    "tram_stops": "city-circle-tram-stops",
    "train_stations": "metro-train-stations-with-accessibility-
information",
    "pedestrians": "pedestrian-counting-system-monthly-counts-per-hour"

}

#datasets
landmarks_df = API_Unlimited(datasets['landmarks'], apikey)
bus_stops_df = API_Unlimited(datasets['bus_stops'], apikey)
tram_stops_df = API_Unlimited(datasets['tram_stops'], apikey)
train_stations_df = API_Unlimited(datasets['train_stations'], apikey)
pedestrian_counts_df = API_Unlimited(datasets['pedestrians'], apikey)
```

```
                              theme  \
       30                    Office
       18                 Mixed Use
       154       Leisure/Recreation
       73        Leisure/Recreation
       20             Community Use
       195         Place Of Assembly
```

```
165          Place of Worship
125             Community Use
85   Residential Accommodation
54                   Transport

                                        sub_theme  \
30                                         Office
18                                   Retail/Office
154                            Gymnasium/Health Club
73   Informal Outdoor Facility (Park/Garden/Reserve)
20                               Government Building
195            Function/Conference/Exhibition Centre
165                                        Church
125                                Public Buildings
85                                 Dwelling (House)
54                                  Railway Station

                                     feature_name  \
30              SBS (Special Broadcasting Service)
18                               Treasury Reserve
154  North Melbourne Recreation Centre (Gymnasium)
73                                   Westgate Park
20                     Melbourne Magistrates Court
195                    Melbourne Convention Centre
165                        St Peter's Eastern Hill
125                              NGV International
85                                    Bishopscourt
54               Melbourne Central Railway Station

                           co_ordinates
30   -37.8176921210305, 144.968740597156
18   -37.8129178793082, 144.975236759799
154  -37.7997051197931, 144.940368286395
73   -37.8314918578874, 144.908824792698
20   -37.8136147671606, 144.956846193891
195  -37.8249040484653, 144.952288281644
165  -37.8097086714637, 144.975259178609
125  -37.8230135000869, 144.969342535451
85    -37.8132921649416, 144.98350690799
54   -37.8108930047401, 144.963100728702
                                geo_point_2d  \
293   -37.78737016259562, 144.96918092237397
8    -37.837547087144706, 144.98191138368836
30    -37.82480198399865, 144.97076232908503
308    -37.818314889062094, 144.956839508202
289   -37.81105987177411, 144.95869339408225
109   -37.78077459328419, 144.95138857277198
45     -37.79443959174042, 144.9295031556217
243   -37.803343440196116, 144.9693670992385
273    -37.80282843793904, 144.9479395483275
135   -37.80111524772101, 144.96674878780823

                                                geo_shape  prop_id
```

```
addresspt1  \
293  {"coordinates": [144.96918092237397, -37.78737...        0
0.000000
8    {"coordinates": [144.98191138368836, -37.83754...        0
41.441167
30   {"coordinates": [144.97076232908503, -37.82480...        0
26.353383
308  {"coordinates": [144.956839508202, -37.8183148...        0
35.877984
289  {"coordinates": [144.95869339408225, -37.81105...        0
31.787580
109  {"coordinates": [144.95138857277198, -37.78077...   107426
55.825150
45   {"coordinates": [144.9295031556217, -37.794439...        0
2.826674
243  {"coordinates": [144.9693670992385, -37.803343...        0
10.914450
273  {"coordinates": [144.9479395483275, -37.802828...        0
13.532624
135  {"coordinates": [144.96674878780823, -37.80111...        0
5.228496

     addressp_1 asset_clas               asset_type  objectid
str_id  \
293           0    Signage  Sign - Public Transport     39748
1252536
8            78    Signage  Sign - Public Transport      2922
1248743
30          200    Signage  Sign - Public Transport     15210
1239404
308         285    Signage  Sign - Public Transport     44101
1268402
289         239    Signage  Sign - Public Transport     36816
1252743
109         306    Signage  Sign - Public Transport      7176
1244570
45          299    Signage  Sign - Public Transport     23024
1577042
243         117    Signage  Sign - Public Transport     16758
1240396
273         123    Signage  Sign - Public Transport     29192
1251207
135         179    Signage  Sign - Public Transport     17860
1240314

     addresspt  asset_subt                       model_desc   mcc_id
\
293          0         NaN  Sign - Public Transport 1 Panel  1252536
8       107419         NaN  Sign - Public Transport 1 Panel  1248743
30      540076         NaN  Sign - Public Transport 1 Panel  1239404
308     105393         NaN  Sign - Public Transport 1 Panel  1268402
289     577288         NaN  Sign - Public Transport 1 Panel  1252743
109     111342         NaN  Sign - Public Transport 1 Panel  1244570
```

```
45      106320         NaN  Sign - Public Transport 1 Panel  1577042
243     612989         NaN  Sign - Public Transport 1 Panel  1240396
273     107985         NaN  Sign - Public Transport 1 Panel  1251207
135     106109         NaN  Sign - Public Transport 1 Panel  1240314

     roadseg_id                                        descriptio
model_no
293       22508  Sign - Public Transport 1 Panel Bus Stop Type 8
P.16
8         22245  Sign - Public Transport 1 Panel Bus Stop Type 8
P.16
30        22466  Sign - Public Transport 1 Panel Bus Stop Type 8
P.16
308       20118  Sign - Public Transport 1 Panel Bus Stop Type 8
P.16
289       20026  Sign - Public Transport 1 Panel Bus Stop Type 8
P.16
109           0  Sign - Public Transport 1 Panel Bus Stop Type 8
P.16
45        21693  Sign - Public Transport 1 Panel Bus Stop Type 1
P.16
243       20556  Sign - Public Transport 1 Panel Bus Stop Type 3
P.16
273       21015  Sign - Public Transport 1 Panel Bus Stop Type 8
P.16
135       20530  Sign - Public Transport 1 Panel Bus Stop Type 3
P.16
                              geo_point_2d  \
10   -37.82097269970027, 144.95546153614245
12  -37.811771476718356, 144.95644059700524
27   -37.81667338583987, 144.97015587085124
18    -37.818324403770184, 144.964479208357
3   -37.813414856197724, 144.94137823870162
23  -37.810354377085794, 144.96136850025573
20   -37.81865571347738, 144.94650837136655
9    -37.82023778673241, 144.95786314283018
15  -37.808876998255194, 144.96634474519394
6     -37.8081489607039, 144.96879323779422

                                          geo_shape  \
10  {"coordinates": [144.95546153614245, -37.82097...
12  {"coordinates": [144.95644059700524, -37.81177...
27  {"coordinates": [144.97015587085124, -37.81667...
18  {"coordinates": [144.964479208357, -37.8183244...
3   {"coordinates": [144.94137823870162, -37.81341...
23  {"coordinates": [144.96136850025573, -37.81035...
20  {"coordinates": [144.94650837136655, -37.81865...
9   {"coordinates": [144.95786314283018, -37.82023...
15  {"coordinates": [144.96634474519394, -37.80887...
6   {"coordinates": [144.96879323779422, -37.80814...

                                    name      xorg stop_no  mccid_str
\
```

```
10      Spencer Street / Flinders Street  GIS Team       1        NaN
12      William Street / La Trobe Street  GIS Team       3        NaN
27      Russell Street / Flinders Street  GIS Team       6        NaN
18    Elizabeth Street / Flinders Street  GIS Team       4        NaN
3   New Quay Promenade / Docklands Drive  GIS Team     D10        NaN
23    Elizabeth Street / La Trobe Street  GIS Team       5        NaN
20     Bourke Street / Harbour Esplanade  GIS Team      D3        NaN
9   Melbourne Aquarium / Flinders Street  GIS Team       2        NaN
15      Russell Street / La Trobe Street  GIS Team       7        NaN
6    Exhibition Street / La Trobe Street  GIS Team       8        NaN

    xsource       xdate  mccid_int
10  Mapbase  2011-10-18          5
12  Mapbase  2011-10-18         16
27  Mapbase  2011-10-18         28
18  Mapbase  2011-10-18          2
3   Mapbase  2011-10-18         11
23  Mapbase  2011-10-18         18
20  Mapbase  2011-10-18          9
9   Mapbase  2011-10-18          4
15  Mapbase  2011-10-18         20
6   Mapbase  2011-10-18         21
                                 geo_point_2d  \
138  -37.815968999999996, 145.22881600000005
139           -37.83172399999995, 145.069614
40     -37.91071299999999, 145.03822300000002
26     -37.88912799999997, 145.04222000000004
142    -38.10435799999999, 145.12823300000002
54               -37.95669399999997, 145.163
140    -37.74287199999998, 145.06611900000007
7      -38.03968099999997, 145.34497299999998
103  -37.843763999999965, 145.07541700000002
114          -37.809699999999964, 144.963757

                                              geo_shape he_loop lift
\
138  {"coordinates": [145.22881600000005, -37.81596...     Yes  Yes
139  {"coordinates": [145.069614, -37.8317239999999...      No   No
40   {"coordinates": [145.03822300000002, -37.91071...     Yes  Yes
26   {"coordinates": [145.04222000000004, -37.88912...      No   No
142  {"coordinates": [145.12823300000002, -38.10435...      No   No
54   {"coordinates": [145.163, -37.95669399999997],...      No   No
140  {"coordinates": [145.06611900000007, -37.74287...      No   No
7    {"coordinates": [145.34497299999998, -38.03968...      No   No
103  {"coordinates": [145.07541700000002, -37.84376...      No   No
114  {"coordinates": [144.963757, -37.8096999999999...     Yes  Yes

          pids            station
138  Dot Matrix          Ringwood
139  Dot Matrix        Riversdale
40          LCD          Mckinnon
26   Dot Matrix        Glenhuntly
142  Dot Matrix           Seaford
```

```
54           No       Sandown Park
140  Dot Matrix            Rosanna
7    Dot Matrix            Berwick
103  Dot Matrix           Hartwell
114         LCD  Melbourne Central
                    id  location_id sensing_date  hourday
direction_1  \
453957    182320240322           18   2024-03-22       23
5
305049    611420220801           61   2022-08-01       14
290
715639    672120220731           67   2022-07-31       21
41
805542    301920230809           30   2023-08-09       19
253
446513     70520231006           70   2023-10-06        5
0
268386    202320240825           20   2024-08-25       23
25
1827901   181720230611           18   2023-06-11       17
55
1408991  1372020240613          137   2024-06-13       20
1
51870     351520221104           35   2022-11-04       15
1039
379811     75720211119           75   2021-11-19        7
23

         direction_2  pedestriancount   sensor_name  \
453957             4                9       Col12_T
305049           459              749      RMIT14_T
715639            63              104      FLDegS_T
805542           394              647      Lon189_T
446513             5                5     Errol20_T
268386            47               72      LtB170_T
1827901           51              106       Col12_T
1408991            2                3  BouHbr2353_T
51870           1170             2209      SouthB_T
379811            14               37      SprFli_T

                          location
453957   -37.81344862, 144.97305353
305049   -37.80767455, 144.96309114
715639   -37.81688755, 144.96562569
805542    -37.8112185, 144.96656806
446513   -37.80456984, 144.94946228
268386    -37.81172914, 144.9682466
1827901  -37.81344862, 144.97305353
1408991  -37.81894815, 144.94612292
51870    -37.82017828, 144.96508877
379811   -37.81515276, 144.97467661
```

Data Preparation and Cleaning

In [3]:
```
#Function to display basic information about a DataFrame
def explore_data(df, name):
    print(f"\n{name} DataFrame Info:")
    print(df.info())
    print(f"\n{name} DataFrame Description:")
    print(df.describe())
    print(f"\n{name} DataFrame Head:")
    print(df.head())
    print(f"\n{name} DataFrame Sample:")
    print(df.sample(5))

# Explore each dataset
if landmarks_df is not None:
    explore_data(landmarks_df, "Landmarks")

if bus_stops_df is not None:
    explore_data(bus_stops_df, "Bus Stops")

if tram_stops_df is not None:
    explore_data(tram_stops_df, "Tram Stops")

if train_stations_df is not None:
    explore_data(train_stations_df, "Train Stations")
```

```
Landmarks DataFrame Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 242 entries, 0 to 241
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   theme         242 non-null    object
 1   sub_theme     242 non-null    object
 2   feature_name  242 non-null    object
 3   co_ordinates  242 non-null    object
dtypes: object(4)
memory usage: 7.7+ KB
None

Landmarks DataFrame Description:
                    theme
sub_theme  \
count                 242
242
unique                 16
```

```
49
top     Leisure/Recreation  Informal Outdoor Facility
(Park/Garden/Reserve)
freq                    63
37

                            feature_name
co_ordinates
count                                242
242
unique                               233
242
top     Melbourne Central Railway Station  -37.8118847831837,
144.962422614541
freq                                   4
1

Landmarks DataFrame Head:
              theme sub_theme                 feature_name  \
0  Place of Worship    Church            St Francis Church
1  Place of Worship    Church              St James Church
2  Place of Worship    Church    St Mary's Anglican Church
3  Place of Worship    Church                 Scots Church
4  Place of Worship    Church  St Michael's Uniting Church

                         co_ordinates
0  -37.8118847831837, 144.962422614541
1  -37.8101281201969, 144.952468571683
2  -37.8031663672997, 144.953761537074
3   -37.8145687802664, 144.96855105335
4  -37.8143851324913, 144.969174036096

Landmarks DataFrame Sample:
               theme              sub_theme
feature_name  \
88         Mixed Use  Retail/Office/Carpark             Council House
2 (CH2)
49         Transport        Railway Station        Royal Park Railway
Station
47         Transport        Railway Station  South Kensington Railway
Station
0   Place of Worship                Church                 St
Francis Church
84   Health Services       Private Hospital       Epworth Freemasons
Hospital

                          co_ordinates
88  -37.8142591432011, 144.966638432727
49  -37.7812684282825, 144.951667833921
47  -37.7997270537735, 144.925837749713
0   -37.8118847831837, 144.962422614541
84   -37.8109710654051, 144.98370007659
```

```
Bus Stops DataFrame Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 309 entries, 0 to 308
Data columns (total 16 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   geo_point_2d  309 non-null    object
 1   geo_shape     309 non-null    object
 2   prop_id       309 non-null    int64
 3   addresspt1    309 non-null    float64
 4   addressp_1    309 non-null    int64
 5   asset_clas    309 non-null    object
 6   asset_type    309 non-null    object
 7   objectid      309 non-null    int64
 8   str_id        309 non-null    int64
 9   addresspt     309 non-null    int64
 10  asset_subt    0 non-null      float64
 11  model_desc    309 non-null    object
 12  mcc_id        309 non-null    int64
 13  roadseg_id    309 non-null    int64
 14  descriptio    309 non-null    object
 15  model_no      309 non-null    object
dtypes: float64(2), int64(7), object(7)
memory usage: 38.8+ KB
None

Bus Stops DataFrame Description:
             prop_id  addresspt1  addressp_1      objectid
str_id  \
count     309.000000  309.000000  309.000000    309.000000
3.090000e+02
mean     6405.006472   25.802489  175.258900  23327.242718
1.296812e+06
std     58324.056187   20.458442  109.574787  13112.345496
1.110742e+05
min         0.000000    0.000000    0.000000    303.000000
1.231165e+06
25%         0.000000   10.980840   88.000000  12390.000000
1.239533e+06
50%         0.000000   21.561304  175.000000  22943.000000
1.249163e+06
75%         0.000000   35.066244  268.000000  35532.000000
1.257190e+06
max    627016.000000   98.326608  360.000000  44401.000000
1.581811e+06

           addresspt  asset_subt        mcc_id    roadseg_id
count     309.000000         0.0  3.090000e+02    309.000000
mean   345929.582524         NaN  1.296812e+06  21305.511327
std    243928.214019         NaN  1.110742e+05   3107.476239
min         0.000000         NaN  1.231165e+06      0.000000
25%    107520.000000         NaN  1.239533e+06  20563.000000
50%    511185.000000         NaN  1.249163e+06  21680.000000
```

```
75%    568579.000000         NaN  1.257190e+06  22386.000000
max    664059.000000         NaN  1.581811e+06  30708.000000

Bus Stops DataFrame Head:
                             geo_point_2d  \
0  -37.80384165792465, 144.93239283833262
1   -37.81548699581418, 144.9581794249902
2  -37.81353897396532, 144.95728334230756
3  -37.82191394843844, 144.95539345270072
4  -37.83316401267591, 144.97443745130263

                                          geo_shape  prop_id
addresspt1  \
0  {"coordinates": [144.93239283833262, -37.80384...        0
76.819824
1  {"coordinates": [144.9581794249902, -37.815486...        0
21.561304
2  {"coordinates": [144.95728334230756, -37.81353...        0
42.177187
3  {"coordinates": [144.95539345270072, -37.82191...        0
15.860434
4  {"coordinates": [144.97443745130263, -37.83316...        0
0.000000

   addressp_1 asset_clas              asset_type  objectid   str_id
\
0         357    Signage  Sign - Public Transport       355  1235255
1          83    Signage  Sign - Public Transport       600  1231226
2         207    Signage  Sign - Public Transport       640  1237092
3         181    Signage  Sign - Public Transport       918  1232777
4           0    Signage  Sign - Public Transport      1029  1271914

   addresspt  asset_subt                       model_desc   mcc_id  \
0     570648         NaN  Sign - Public Transport 1 Panel  1235255
1     548056         NaN  Sign - Public Transport 1 Panel  1231226
2     543382         NaN  Sign - Public Transport 1 Panel  1237092
3     103975         NaN  Sign - Public Transport 1 Panel  1232777
4          0         NaN  Sign - Public Transport 1 Panel  1271914

   roadseg_id                                       descriptio
model_no
0       21673  Sign - Public Transport 1 Panel Bus Stop Type 13
P.16
1       20184   Sign - Public Transport 1 Panel Bus Stop Type 8
P.16
2       20186   Sign - Public Transport 1 Panel Bus Stop Type 8
P.16
3       22174   Sign - Public Transport 1 Panel Bus Stop Type 8
P.16
4       22708   Sign - Public Transport 1 Panel Bus Stop Type 8
P.16

Bus Stops DataFrame Sample:
```

```
                                geo_point_2d  \
282    -37.82093926515732, 144.9271057669481
230   -37.82190993188825, 144.93242660517498
181  -37.807560935966926, 144.91815632059857
110  -37.811942864580594, 144.95642939829426
22    -37.80936279878404, 144.95558831833387

                                           geo_shape  prop_id
addresspt1  \
282  {"coordinates": [144.9271057669481, -37.820939...        0
23.506067
230  {"coordinates": [144.93242660517498, -37.82190...   561106
12.039283
181  {"coordinates": [144.91815632059857, -37.80756...        0
14.679846
110  {"coordinates": [144.95642939829426, -37.81194...        0
20.736365
22   {"coordinates": [144.95558831833387, -37.80936...        0
41.782509

     addressp_1 asset_clas               asset_type  objectid
str_id  \
282         266    Signage  Sign - Public Transport     33197
1245217
230         167    Signage  Sign - Public Transport     12063
1257294
181          75    Signage  Sign - Public Transport     37840
1236357
110          26    Signage  Sign - Public Transport      7511
1237106
22           65    Signage  Sign - Public Transport     12010
1448084

     addresspt  asset_subt                       model_desc   mcc_id
\
282     565798         NaN  Sign - Public Transport 1 Panel  1245217
230     561106         NaN  Sign - Public Transport 1 Panel  1257294
181     514440         NaN  Sign - Public Transport 1 Panel  1236357
110     110759         NaN  Sign - Public Transport 1 Panel  1237106
22      514908         NaN  Sign - Public Transport 1 Panel  1448084

     roadseg_id                                        descriptio
model_no
282       22158   Sign - Public Transport 1 Panel Bus Stop Type 3
P.16
230           0   Sign - Public Transport 1 Panel Bus Stop Type 3
P.16
181       22440  Sign - Public Transport 1 Panel Bus Stop Type 11
P.16
110       20187   Sign - Public Transport 1 Panel Bus Stop Type 8
P.16
22        21528   Sign - Public Transport 1 Panel Bus Stop Type 3
P.16
```

```
Tram Stops DataFrame Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 28 entries, 0 to 27
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   geo_point_2d  28 non-null     object
 1   geo_shape     28 non-null     object
 2   name          28 non-null     object
 3   xorg          28 non-null     object
 4   stop_no       28 non-null     object
 5   mccid_str     0 non-null      float64
 6   xsource       28 non-null     object
 7   xdate         28 non-null     object
 8   mccid_int     28 non-null     int64
dtypes: float64(1), int64(1), object(7)
memory usage: 2.1+ KB
None

Tram Stops DataFrame Description:
       mccid_str  mccid_int
count        0.0  28.000000
mean         NaN  14.500000
std          NaN   8.225975
min          NaN   1.000000
25%          NaN   7.750000
50%          NaN  14.500000
75%          NaN  21.250000
max          NaN  28.000000

Tram Stops DataFrame Head:
                              geo_point_2d  \
0    -37.81922319307822, 144.9614014008424
1  -37.821539117626855, 144.95356912978238
2  -37.815426586135686, 144.94512063442602
3  -37.813414856197724, 144.94137823870162
4  -37.814591782869805, 144.94655055842398

                                          geo_shape  \
0  {"coordinates": [144.9614014008424, -37.819223...
1  {"coordinates": [144.95356912978238, -37.82153...
2  {"coordinates": [144.94512063442602, -37.81542...
3  {"coordinates": [144.94137823870162, -37.81341...
4  {"coordinates": [144.94655055842398, -37.81459...

                                    name      xorg stop_no
mccid_str  \
0           Market Street / Flinders Street  GIS Team       3
NaN
1  Victoria Police Centre / Flinders Street  GIS Team      D6
NaN
2          Central Pier / Harbour Esplanade  GIS Team      D2
```

```
NaN
3      New Quay Promenade / Docklands Drive  GIS Team     D10
NaN
4          Etihad Statium / La Trobe Street  GIS Team      D1
NaN

   xsource       xdate  mccid_int
0  Mapbase  2011-10-18          3
1  Mapbase  2011-10-18          6
2  Mapbase  2011-10-18         10
3  Mapbase  2011-10-18         11
4  Mapbase  2011-10-18         13

Tram Stops DataFrame Sample:
                               geo_point_2d  \
5   -37.812487918072826, 144.95393532833103
25    -37.80801062631204, 144.9731036500197
6     -37.8081489607039, 144.96879323779422
12  -37.811771476718356, 144.95644059700524
21    -37.814464564872125, 144.938645561825

                                          geo_shape  \
5   {"coordinates": [144.95393532833103, -37.81248...
25  {"coordinates": [144.9731036500197, -37.808010...
6   {"coordinates": [144.96879323779422, -37.80814...
12  {"coordinates": [144.95644059700524, -37.81177...
21  {"coordinates": [144.938645561825, -37.8144645...

                                  name      xorg stop_no  mccid_str
xsource  \
5         King Street / La Trobe Street  GIS Team       2        NaN
Mapbase
25   Nicholson Street / Victoria Parade  GIS Team      10        NaN
Mapbase
6   Exhibition Street / La Trobe Street  GIS Team       8        NaN
Mapbase
12     William Street / La Trobe Street  GIS Team       3        NaN
Mapbase
21    Waterfront City / Docklands Drive  GIS Team     D11        NaN
Mapbase

         xdate  mccid_int
5   2011-10-18         15
25  2011-10-18         23
6   2011-10-18         21
12  2011-10-18         16
21  2011-10-18         12

Train Stations DataFrame Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 219 entries, 0 to 218
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype
```

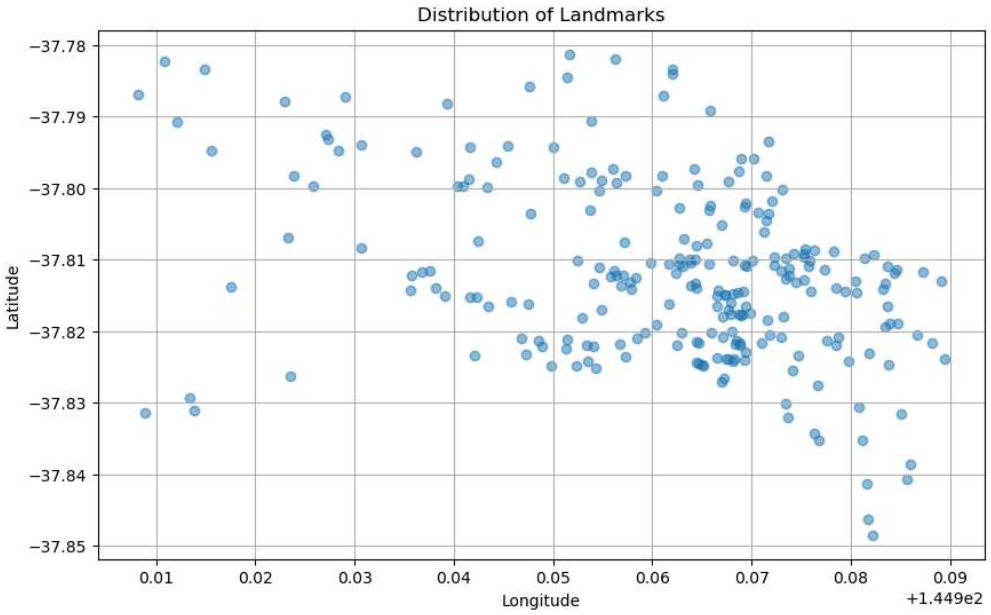
```
---  ------        --------------  -----
 0   geo_point_2d  219 non-null    object
 1   geo_shape     219 non-null    object
 2   he_loop       219 non-null    object
 3   lift          219 non-null    object
 4   pids          219 non-null    object
 5   station       219 non-null    object
dtypes: object(6)
memory usage: 10.4+ KB
None

Train Stations DataFrame Description:
                          geo_point_2d  \
count                              219
unique                             219
top     -37.77839599999999, 145.031251
freq                                 1

                                                 geo_shape he_loop
lift  \
count                                                219     219
219
unique                                               219       3
2
top     {"coordinates": [145.031251, -37.7783959999999...      No
No
freq                                                   1     190
185

              pids     station
count          219         219
unique           4         219
top     Dot Matrix  Alphington
freq           115           1

Train Stations DataFrame Head:
                             geo_point_2d  \
0           -37.77839599999999, 145.031251
1           -37.86724899999996, 144.830604
2  -37.761897999999974, 144.96056099999998
3           -37.82241099999999, 145.045617
4   -37.73345899999998, 144.96274700000004

                                        geo_shape he_loop lift
pids  \
0  {"coordinates": [145.031251, -37.7783959999999...      No   No
Dot Matrix
1  {"coordinates": [144.830604, -37.8672489999999...      No   No
LCD
2  {"coordinates": [144.96056099999998, -37.76189...      No   No
No
3  {"coordinates": [145.045617, -37.8224109999999...      No   No
No
```

```
4  {"coordinates": [144.96274700000004, -37.73345...      No    No
No

      station
0  Alphington
1      Altona
2      Anstey
3      Auburn
4      Batman

Train Stations DataFrame Sample:
                                 geo_point_2d  \
86   -37.966943999999955, 145.05477200000007
43             -37.84271899999999, 144.883604
124                     -37.700717, 144.77386
117  -37.915134999999964, 144.99629900000002
144   -37.90369199999998, 145.33133500000008

                                           geo_shape he_loop lift
\
86   {"coordinates": [145.05477200000007, -37.96694...      No   No
43   {"coordinates": [144.883604, -37.8427189999999...      No   No
124  {"coordinates": [144.77386, -37.700717], "type...      No  Yes
117  {"coordinates": [144.99629900000002, -37.91513...      No   No
144  {"coordinates": [145.33133500000008, -37.90369...      No   No

           pids          station
86   Dot Matrix       Cheltenham
43          LCD          Newport
124  Dot Matrix     Watergardens
117          No  Middle Brighton
144          No            Upwey
```

In [4]:
```
#using Landmarks DataFrame to parse coordinates
def parse_coordinates(df, coord_col):
    #Splitting the 'co_ordinates' column into two columns
    df[['latitude', 'longitude']] = df[coord_col].str.split(', ',
expand=True)
    df['latitude'] = pd.to_numeric(df['latitude'])
    df['longitude'] = pd.to_numeric(df['longitude'])
    return df

landmarks_df = parse_coordinates(landmarks_df, 'co_ordinates')
print(landmarks_df.head())
```

```
                theme sub_theme                 feature_name  \
0  Place of Worship    Church            St Francis Church
1  Place of Worship    Church              St James Church
2  Place of Worship    Church    St Mary's Anglican Church
3  Place of Worship    Church                 Scots Church
4  Place of Worship    Church  St Michael's Uniting Church

                          co_ordinates   latitude   longitude
0  -37.8118847831837, 144.962422614541 -37.811885  144.962423
1  -37.8101281201969, 144.952468571683 -37.810128  144.952469
2  -37.8031663672997, 144.953761537074 -37.803166  144.953762
3   -37.8145687802664, 144.96855105335 -37.814569  144.968551
4  -37.8143851324913, 144.969174036096 -37.814385  144.969174
```

In [5]:
```
def plot_geographic_data(df, title):
    plt.figure(figsize=(10, 6))
    plt.scatter(df['longitude'], df['latitude'], alpha=0.5)
    plt.title(title)
    plt.xlabel('Longitude')
    plt.ylabel('Latitude')
    plt.grid(True)
    plt.show()

#Plotting data for landmarks
plot_geographic_data(landmarks_df, 'Distribution of Landmarks')
```



Description of the Distribution of Landmarks in Melbourne

The scatter plot above illustrates the geographical distribution of landmarks across Melbourne. Each point represents a place of interest, such as schools, theatres, health services, sports venues, and other public facilities. These landmarks are plotted based on their latitude and longitude coordinates.

Key Observations:

Concentration of Landmarks:
The majority of landmarks are densely clustered in the middle-right section of the plot. This suggests that a significant number of landmarks are concentrated around specific areas of Melbourne, which could be central locations with more public facilities or higher population densities.

Outliers:
There are several outliers scattered across the plot, representing landmarks located farther away from the central cluster. These could be facilities situated in less populated or more suburban parts of Melbourne.
Latitude and Longitude Range:
The latitude values range between -37.78 and -37.85, which covers a significant portion of Melbourne's area. The longitude values are presented relative to 144.92 degrees east, which places the landmarks within the general vicinity of Melbourne's city center.

General Distribution:
The distribution seems more densely populated between latitude -37.80 and -37.83, indicating areas where key places of interest are more likely to be located.

In [6]:
```
def parse_coordinates(df, coord_col):
    df[['latitude', 'longitude']] = df[coord_col].str.split(', ',
expand=True)
    df['latitude'] = pd.to_numeric(df['latitude'])
    df['longitude'] = pd.to_numeric(df['longitude'])
    return df
bus_stops_df = parse_coordinates(bus_stops_df, 'geo_point_2d')
print(bus_stops_df[['latitude', 'longitude']].head())
```

```
    latitude   longitude
0 -37.803842  144.932393
1 -37.815487  144.958179
2 -37.813539  144.957283
3 -37.821914  144.955393
4 -37.833164  144.974437
```

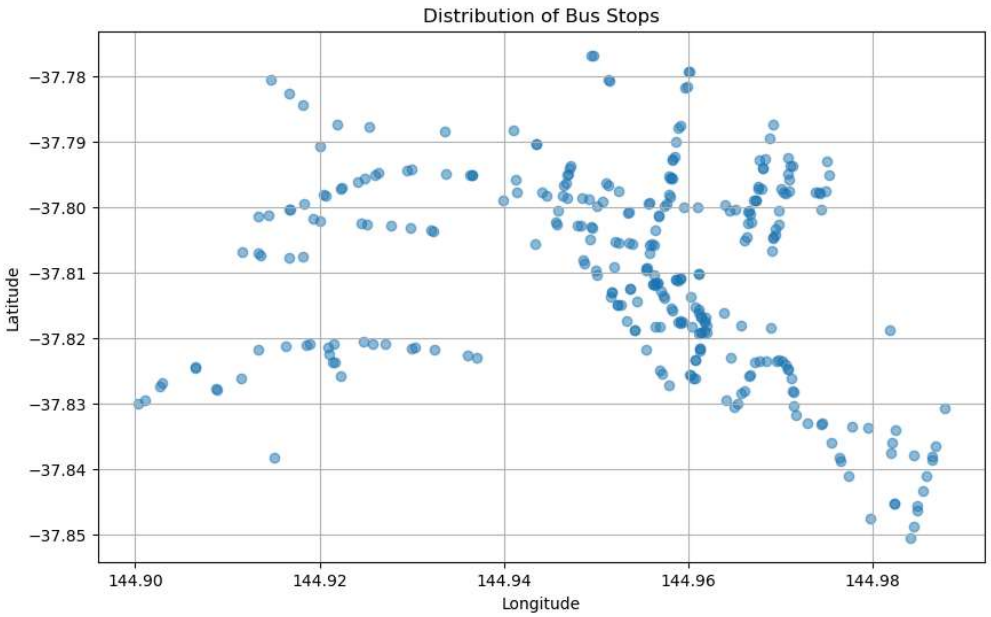
In [7]:
```
bus_stops_df.drop(columns=['asset_subt'], inplace=True)
```

In [8]:
```
bus_stops_df.head()
```

Out[8]:

	geo_point_2d	geo_shape	prop_id	addresspt1	ad
0	-37.80384165792465, 144.93239283833262	{"coordinates": [144.93239283833262, -37.80384...	0	76.819824	35
1	-37.81548699581418, 144.9581794249902	{"coordinates": [144.9581794249902, -37.815486...	0	21.561304	83
2	-37.81353897396532, 144.95728334230756	{"coordinates": [144.95728334230756, -37.81353...	0	42.177187	20
3	-37.82191394843844, 144.95539345270072	{"coordinates": [144.95539345270072, -37.82191...	0	15.860434	18
4	-37.83316401267591, 144.97443745130263	{"coordinates": [144.97443745130263, -37.83316...	0	0.000000	0

In [9]:
```
#Plotting data for Bus Stops.
plot_geographic_data(bus_stops_df, 'Distribution of Bus Stops')
```



Distribution of Bus Stops in Melbourne

The scatter plot above depicts the distribution of bus stops in Melbourne, using longitude and latitude coordinates to show where these public transport facilities are located.

Key Observations:

Concentration of Bus Stops:
A significant cluster of bus stops is concentrated in the longitude range of 144.94 to 144.96 and latitude range of -37.80 to -37.83. This suggests a well-serviced area, likely corresponding to a more urban or densely populated region where public transport infrastructure is heavily used.

Spread of Bus Stops:
Bus stops are more sparsely distributed in areas between longitude 144.90 to 144.92 and longitude 144.96 to 144.98. These regions might correspond to suburban or less densely populated areas where fewer bus stops are required due to lower demand.

Patterns and Grid-like Structure:
The distribution of points appears to form a semi-grid-like pattern in some sections, indicating that bus stops might be positioned systematically along main roads or intersections in Melbourne. This could also reflect bus routes that follow regular road networks in the city.

Outliers and Coverage:
There are outlier points, particularly around longitude 144.90 and 144.98, where bus stops are more isolated, possibly serving remote or less frequently visited areas.

Latitude and Longitude Range:
The bus stops cover a region between latitude -37.78 and -37.85 and longitude 144.90 to 144.98, which is consistent with the central and surrounding areas of Melbourne.

In [10]:
```
print(bus_stops_df.isnull().sum())

#Data type verification
print(bus_stops_df.dtypes)
```

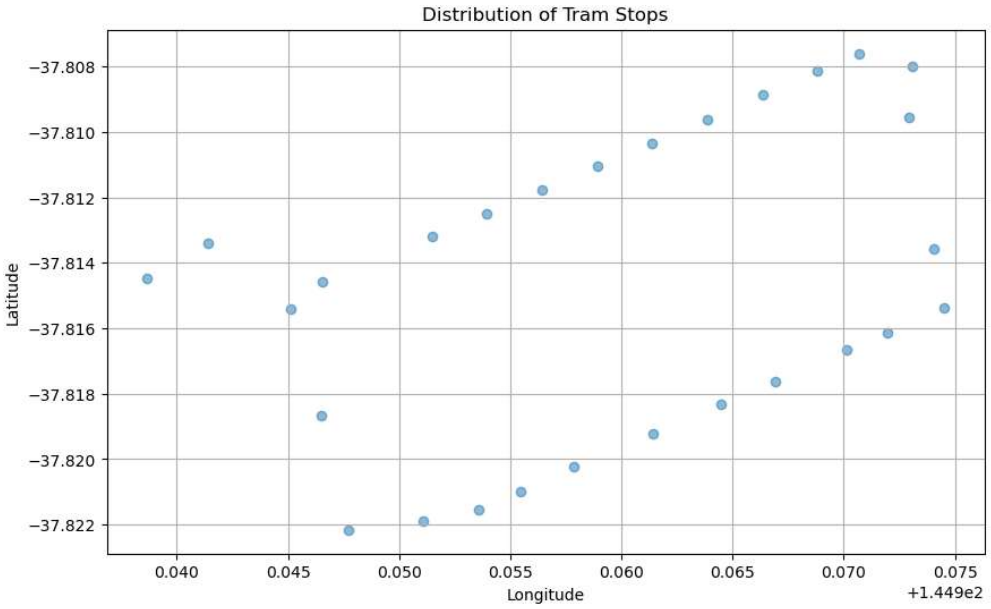
```
geo_point_2d    0
geo_shape       0
prop_id         0
addresspt1      0
addressp_1      0
asset_clas      0
asset_type      0
objectid        0
str_id          0
addresspt       0
model_desc      0
mcc_id          0
roadseg_id      0
descriptio      0
model_no        0
latitude        0
longitude       0
dtype: int64
geo_point_2d     object
geo_shape        object
prop_id           int64
addresspt1      float64
addressp_1        int64
asset_clas       object
asset_type       object
objectid          int64
str_id            int64
addresspt         int64
model_desc       object
mcc_id            int64
roadseg_id        int64
descriptio       object
model_no         object
latitude        float64
longitude       float64
dtype: object
```

In [11]:
```
#Applying the same coordinate parsing function
tram_stops_df = parse_coordinates(tram_stops_df, 'geo_point_2d')
print(tram_stops_df[['latitude', 'longitude']].head())
```

```
    latitude   longitude
0 -37.819223  144.961401
1 -37.821539  144.953569
2 -37.815427  144.945121
3 -37.813415  144.941378
4 -37.814592  144.946551
```

In [12]:
```
tram_stops_df.drop(columns=['mccid_str'], inplace=True)
```

In [13]:
```
plot_geographic_data(tram_stops_df, 'Distribution of Tram Stops')
```



Distribution of Tram Stops in Melbourne

The scatter plot above illustrates the geographical distribution of tram stops in a specific region of Melbourne, using latitude and longitude coordinates.

Key Observations:

Linear Distribution:
The tram stops form a relatively clear linear pattern, indicating that they are aligned along a tram route. This linear arrangement is typical of tram systems, as tram stops tend to follow the path of the tram tracks.

Latitude and Longitude Range:
The latitude values range from approximately -37.822 to -37.808, and the longitude values range from around 144.92 to 144.97. This suggests that the tram stops are located in a specific corridor within Melbourne, possibly representing a central tram line or a major thoroughfare.

Even Spacing:
The stops appear to be evenly spaced, which suggests a planned distribution, likely to ensure tram stops are located at regular intervals along the tram route, providing consistent access to public transport for residents or commuters.

Sparse Distribution Towards Extremes:
As we approach the edges of the plot, particularly around longitude 0.075 and latitude -37.822, the tram stops become sparser. This may indicate the end of a tram line or areas where the density of tram stops decreases, possibly due to fewer passengers or lower demand.

Clustering:
There are areas of mild clustering in the middle of the plot, possibly suggesting higher demand or interchange points along the tram line.

In [16]:
```
print(tram_stops_df.isnull().sum())
#Data type verification
print(tram_stops_df.dtypes)
```

```
geo_point_2d    0
geo_shape       0
name            0
xorg            0
stop_no         0
xsource         0
xdate           0
mccid_int       0
latitude        0
longitude       0
dtype: int64
geo_point_2d     object
geo_shape        object
name             object
xorg             object
stop_no          object
xsource          object
xdate            object
mccid_int         int64
latitude        float64
longitude       float64
dtype: object
```

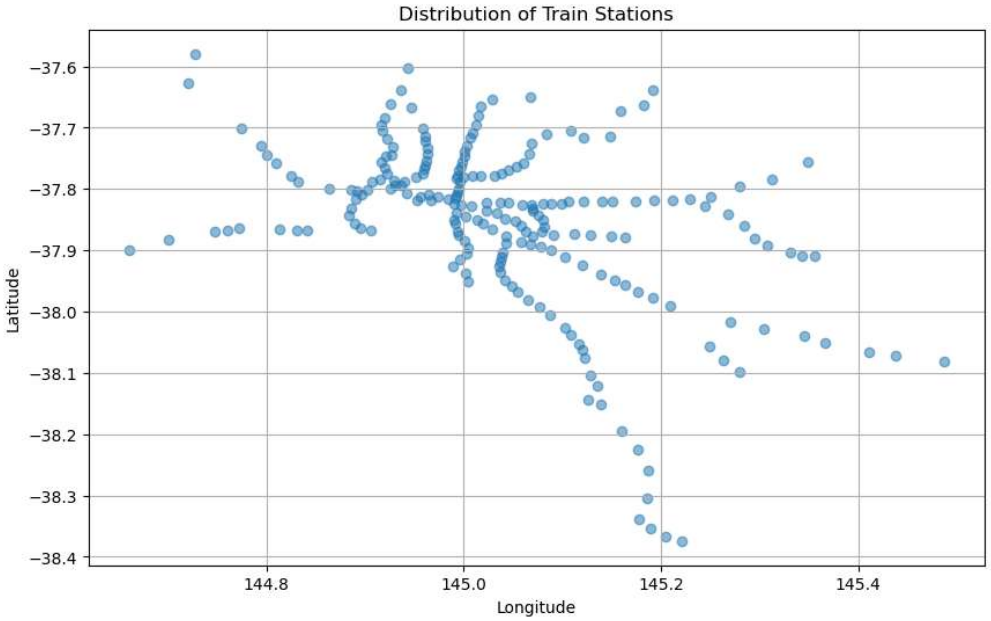
In [17]:
```
#Applying the coordinate parsing function to Train Stations DataFrame
train_stations_df = parse_coordinates(train_stations_df, 'geo_point_2d')
print(train_stations_df[['latitude', 'longitude']].head())
```

```
    latitude   longitude
0 -37.778396  145.031251
1 -37.867249  144.830604
2 -37.761898  144.960561
3 -37.822411  145.045617
4 -37.733459  144.962747
```

In [18]:
```
print(train_stations_df[['he_loop', 'lift', 'pids']].describe())
```

```
       he_loop lift        pids
count      219  219         219
unique       3    2           4
top         No   No  Dot Matrix
freq       190  185         115
```

In [19]:
```
plot_geographic_data(train_stations_df, 'Distribution of Train Stations')
```



Distribution of Train Stations in Melbourne

The scatter plot above represents the distribution of train stations across Melbourne, based on latitude and longitude coordinates.

Key Observations:

Radial Pattern:

The most striking feature of this plot is the radial pattern emanating from a central point near latitude -37.8 and longitude 145.0. This central area likely represents Melbourne's central business district (CBD) or a major transport hub from which multiple train lines extend outward in different directions.

Train Lines:

The spokes of points radiating outward likely represent different train lines serving Melbourne's suburbs. The distribution aligns with the typical structure of a metropolitan rail network, with lines extending from a central hub to outer regions.

Diverse Coverage:

Train stations cover a large geographic area, with latitude values ranging from -37.6 to -38.4 and longitude values ranging from 144.8 to 145.4. This indicates a wide-reaching train system serving both the inner and outer suburbs of Melbourne.

Densely Populated Central Area:

A high density of train stations is concentrated in and around the central region, suggesting this is where public transport demand is greatest. This central cluster likely corresponds to the areas with the highest population density or the most business and commercial activity.

Sparse Outer Coverage:

As you move further away from the central hub, particularly to the north and southeast, the stations become more sparse, reflecting the lower population density in Melbourne's outer suburbs and rural areas.

Gaps in Coverage:

There are noticeable gaps in coverage, particularly in the southeastern and far western regions, which might represent areas with lower accessibility to train services or regions where other forms of public transport (e.g., buses or trams) are more prevalent.

In [20]:
```
print(train_stations_df.isnull().sum())
#Data type verification
print(train_stations_df.dtypes)
```

```
geo_point_2d    0
geo_shape       0
he_loop         0
lift            0
pids            0
station         0
latitude        0
longitude       0
dtype: int64
geo_point_2d     object
geo_shape        object
he_loop          object
lift             object
pids             object
station          object
latitude        float64
longitude       float64
dtype: object
```

Feature Engineering

In [21]:

```
#Creating function to calculate the Haversine distance between two points
on the Earth's surface given their longitude and latitude
def haversine(lon1, lat1, lon2, lat2):
    #Converting decimal degrees to radians
    lon1, lat1, lon2, lat2 = map(np.radians, [lon1, lat1, lon2, lat2])

    #Computing the difference in longitude and latitude between the two
points
    dlon = lon2 - lon1
    dlat = lat2 - lat1

    #Apply the Haversine formula
    a = np.sin(dlat/2)**2 + np.cos(lat1) * np.cos(lat2) * np.sin(dlon/2)**2
    c = 2 * np.arcsin(np.sqrt(a))
    r = 6371  #Radius of Earth in kilometers
    return c * r

#Function to calculate the minimum distances between points in two
dataframes
def calculate_distances(df1, df2):
    min_distances = [] #Initialize an empty list to store the minimum
distances
    for i, row1 in df1.iterrows():
        min_distance = np.inf #Initialize the minimum distance as infinity
to start comparison
        for j, row2 in df2.iterrows():
            distance = haversine(row1['longitude'], row1['latitude'],
row2['longitude'], row2['latitude'])

            #If the calculated distance is smaller than the current minimum
distance, update it
            if distance < min_distance:
                min_distance = distance
        min_distances.append(min_distance) #Append the minimum distance to
the list
    return min_distances

#Calculating nearest distances for each type of transport
landmarks_df['nearest_bus_distance'] = calculate_distances(landmarks_df,
bus_stops_df)
landmarks_df['nearest_tram_distance'] = calculate_distances(landmarks_df,
tram_stops_df)
landmarks_df['nearest_train_distance'] = calculate_distances(landmarks_df,
train_stations_df)
```

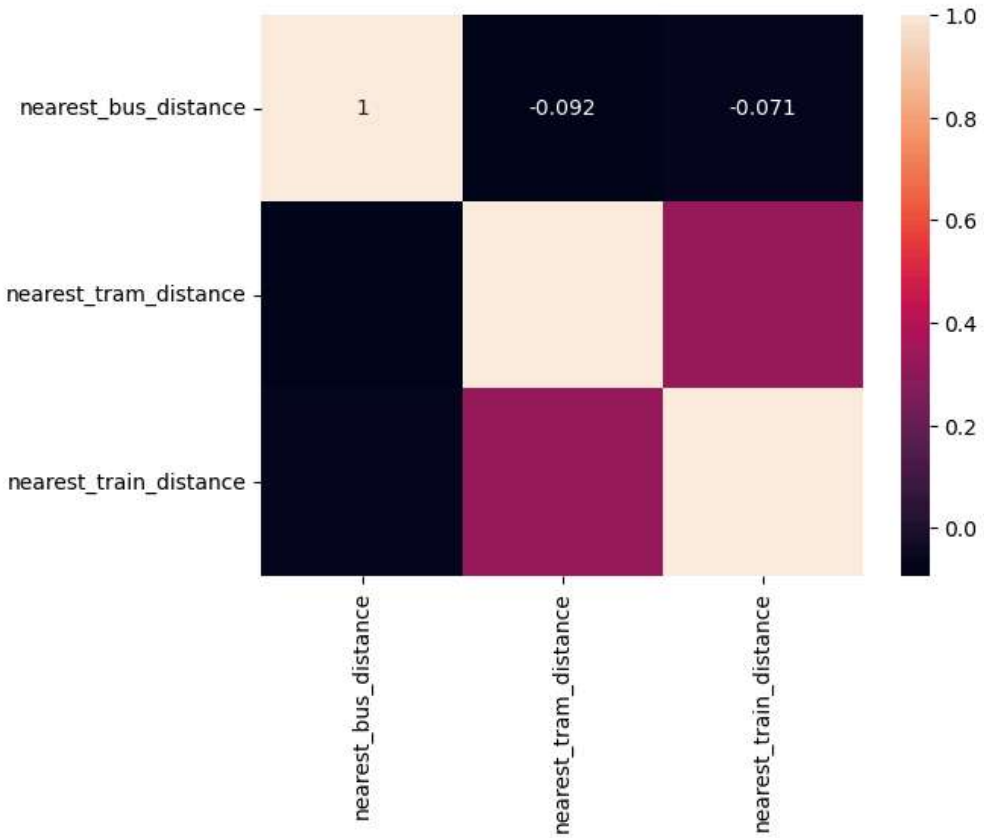
In [22]:
```
#Calculating accessibility index
def calculate_accessibility_index(row):
    weights = {'bus': 0.4, 'tram': 0.4, 'train': 0.2}
    index = (weights['bus'] / (row['nearest_bus_distance'] + 0.1) +  #
Avoid division by zero
             weights['tram'] / (row['nearest_tram_distance'] + 0.1) +
             weights['train'] / (row['nearest_train_distance'] + 0.1))
    return index * 1000  #Scaling factor for better interpretation

landmarks_df['accessibility_index'] =
landmarks_df.apply(calculate_accessibility_index, axis=1)
```

The Accessibility Index is a metric I developed to quantify how well each landmark in Melbourne is served by public transport. It takes into account the proximity of a landmark to the nearest bus stops, tram stops, and train stations. The goal of the Accessibility Index is to provide a single, easy-to-understand score that reflects the overall accessibility of a landmark by public transport.

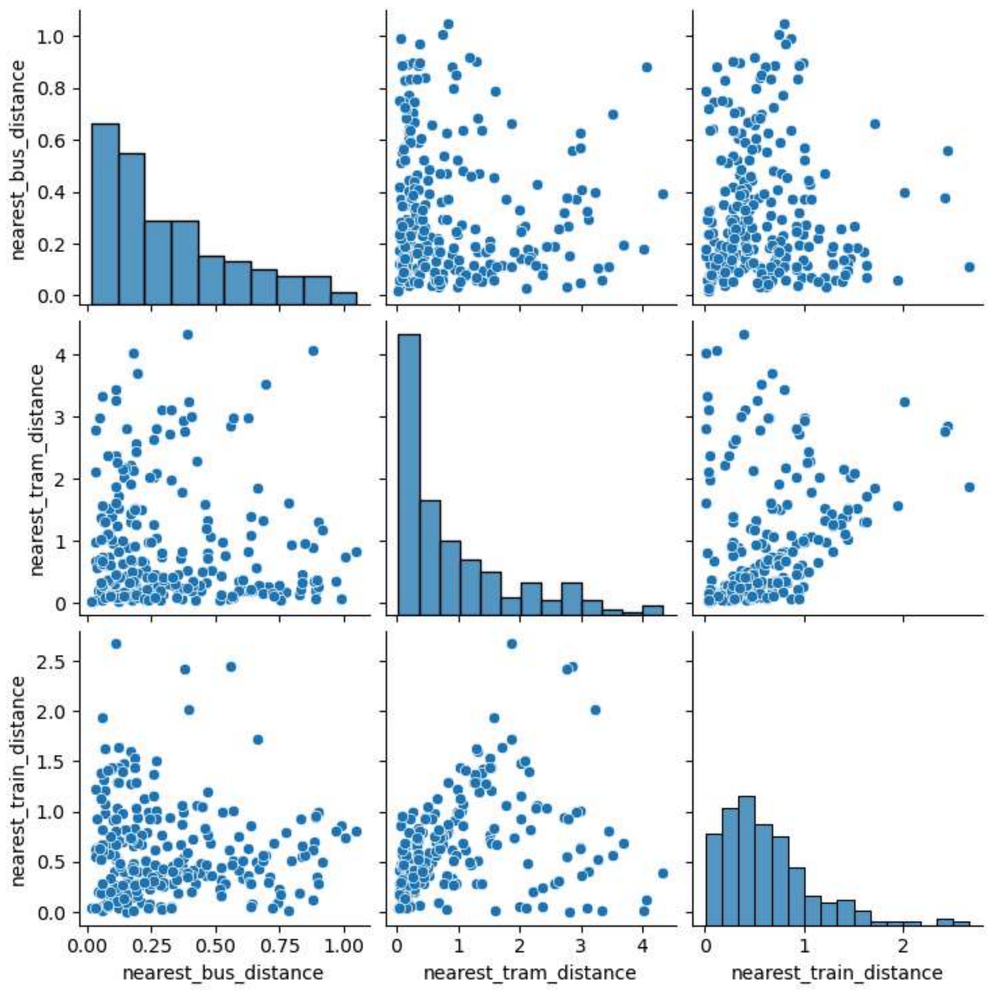
Exploratory Data Analysis (EDA)

Correlation Analysis

In [23]:
```
#Creating a heatmap to visualize the correlation matrix between
'nearest_bus_distance', 'nearest_tram_distance', and
'nearest_train_distance'
sns.heatmap(landmarks_df[['nearest_bus_distance', 'nearest_tram_distance',
'nearest_train_distance']].corr(), annot=True)
plt.show()

#Further Exploratory Data Analysis (EDA) using pair plots
#Creating Pairplot to visualize relationships between
'nearest_bus_distance', 'nearest_tram_distance', and
'nearest_train_distance'
sns.pairplot(landmarks_df, vars=['nearest_bus_distance',
'nearest_tram_distance', 'nearest_train_distance'])
plt.show()
```



```
C:\Users\emman\anaconda3\Lib\site-
packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na
option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\emman\anaconda3\Lib\site-
packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na
option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\emman\anaconda3\Lib\site-
packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na
option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



Analysis of the Correlation Matrix

Heatmap Analysis Observations:

Diagonal Elements:

- Each diagonal value is 1, signifying that each distance measure is perfectly correlated with itself.

Correlation Between Distances:
- **Bus and Tram Distances:** The correlation coefficient is -0.092, indicating an extremely weak inverse relationship. This means that landmarks nearer to bus stops tend to be marginally further from tram stops, though the relationship is minimal.

- **Bus and Train Distances:** The coefficient of -0.071 reflects a similarly weak negative relationship, implying that the proximity of landmarks to bus stops has almost no effect on their distance from train stations.

- **Tram and Train Distances:** A weak positive correlation is observed here. While landmarks closer to tram stops might also be somewhat nearer to train stations, the relationship remains quite weak.

Pair Plot Analysis

Key Observations:

Histograms (Diagonal Elements):

- **Nearest Bus Distance:*** The histogram shows a right-skewed distribution, indicating that most landmarks are relatively close to a bus stop, with fewer landmarks being farther away.
- **Nearest Tram Distance:** Similar to bus distances, the distribution of tram distances is also right-skewed. This suggests that most landmarks have tram stops nearby, with a gradual decrease in frequency as the distance increases.
- **Nearest Train Distance:** The train distance distribution is also right-skewed, implying that most landmarks are relatively close to train stations, though a few are farther away.

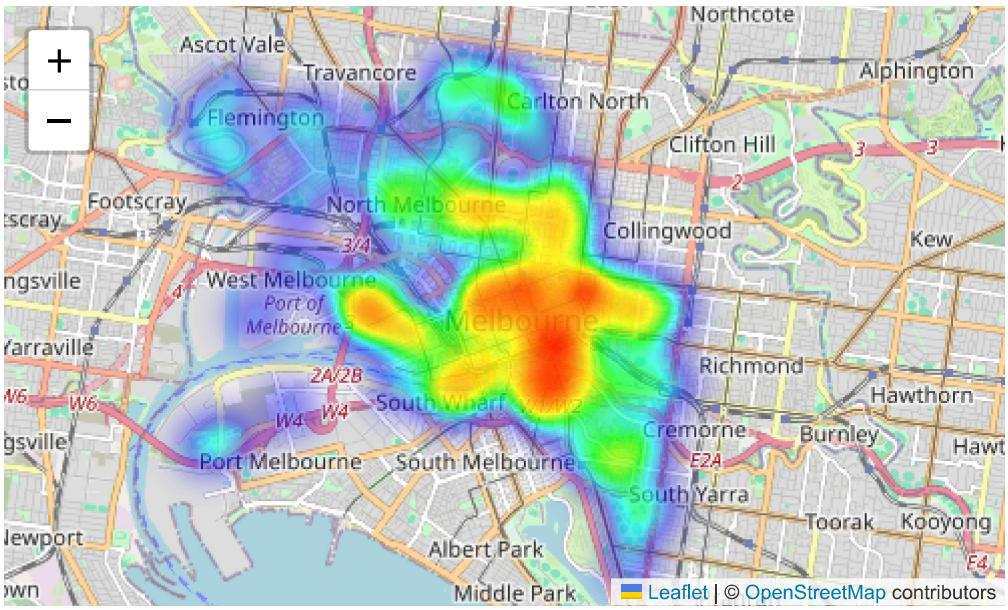
Scatter Plots (Off-Diagonal Elements):

- **Bus vs. Tram Distances:** The scatter plot between nearest bus and tram distances shows a scattered pattern with no strong linear relationship. This suggests that the proximity of a landmark to a bus stop doesn't strongly predict its proximity to a tram stop.
- **Bus vs. Train Distances:** The scatter plot between nearest bus and train distances also shows a weak relationship, indicating that these distances are largely independent of each other.
- **Tram vs. Train Distances:** The relationship between tram and train distances shows a slight positive correlation, particularly when distances are shorter. This implies that landmarks closer to tram stops are somewhat more likely to be near train stations, although the relationship remains weak.

Spatial Analysis

In [29]:
```
# Creating a base map centered on Melbourne's coordinates with a specified
zoom level
map_melbourne = folium.Map(
    location=[-37.8136, 144.9631],  # Melbourne's latitude and longitude
    zoom_start=12  # Initial zoom level for the map (12 is moderately
zoomed in)
)

# Adding a heatmap layer to the map using the latitude, longitude, and
accessibility index from landmarks_df
HeatMap(
    data=landmarks_df[['latitude', 'longitude', 'accessibility_index']],  #
Data used for the heatmap
    radius=15  # Setting the radius of the heatmap points for visualization
).add_to(map_melbourne)  # Adding the heatmap layer to the base map

# Displaying the map directly in the notebook
map_melbourne
```

Out[29]:



The image shows a heatmap centered on Melbourne, highlighting the accessibility of landmarks across different parts of the city. The heatmap is overlaid on a geographic map, with higher intensity colors (green, yellow, and red) representing areas with greater accessibility or concentration of landmarks, while cooler colors (blue and purple) represent areas with fewer landmarks or lower accessibility.

Key Observations:

Concentration of Accessibility:
The most intense areas (in yellow and red) appear around Melbourne's Central Business District (CBD) and adjacent neighborhoods like Southbank and East Melbourne, suggesting these areas have the highest concentration of accessible landmarks.

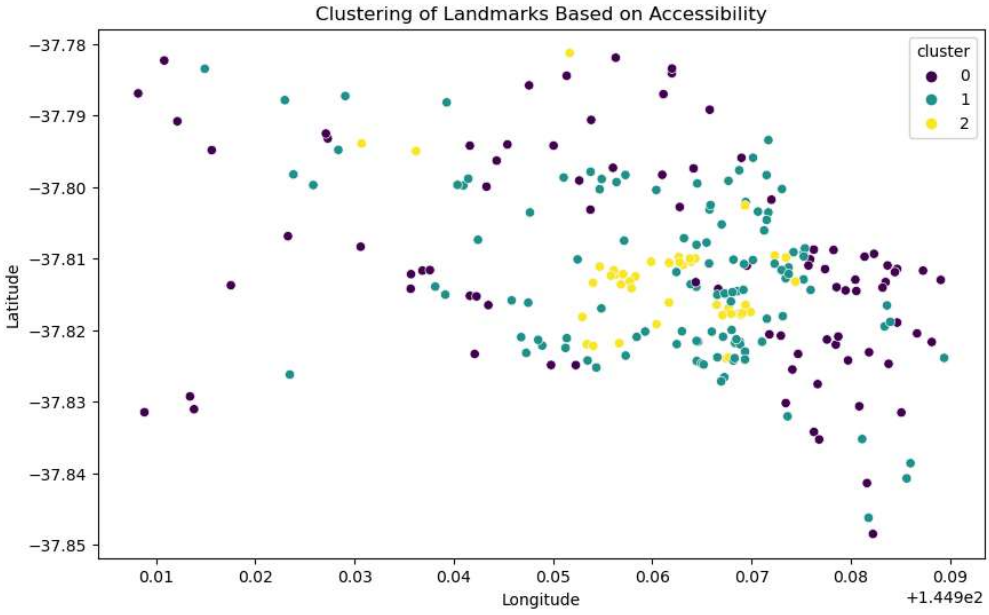
Spreading Influence:
The heatmap also shows moderately accessible areas spreading towards North Melbourne, Carlton, Docklands, and Richmond, indicating that these surrounding areas have a significant number of landmarks but lower accessibility compared to the central regions.

Sparse Areas:
Outlying areas like Kensington, Flemington, and Footscray show lower accessibility levels, as indicated by the cooler colors (blue and purple), suggesting fewer landmarks or less accessibility in these regions.

Cluster Analysis

In [30]:
```
from sklearn.cluster import KMeans

# Applying KMeans clustering
kmeans = KMeans(n_clusters=3, random_state=42)
landmarks_df['cluster'] =
kmeans.fit_predict(landmarks_df[['accessibility_index']])

# Visualize the clusters
plt.figure(figsize=(10, 6))
sns.scatterplot(data=landmarks_df, x='longitude', y='latitude',
hue='cluster', palette='viridis')
plt.title('Clustering of Landmarks Based on Accessibility')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```

```
C:\Users\emman\anaconda3\Lib\site-
packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The
default value of `n_init` will change from 10 to 'auto' in 1.4.
Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
C:\Users\emman\anaconda3\Lib\site-
packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is
known to have a memory leak on Windows with MKL, when there are
less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```



My Analysis of the Clustering Visualization

Clustering Insights:

Clusters Identified: Using the KMeans algorithm, I successfully grouped the landmarks into three distinct clusters based on their accessibility indices. Each cluster is represented by a different color on the scatter plot.

- **Cluster 0 (Purple)**: This group likely represents landmarks with lower accessibility, which could indicate that they are less served by public transport.

- **Cluster 1 (Green)**: These landmarks seem to have moderate accessibility, falling between the extremes.

- **Cluster 2 (Yellow)**:This cluster likely includes the best-connected landmarks, indicating they are well-served by public transport.

Geographic Distribution:

Visual Spread: The scatter plot allows me to see how these clusters are geographically spread out across the area. This helps me identify which regions are well-covered by public transport (as shown by the yellow cluster) and which might need further improvements (as indicated by the purple cluster).

In [31]:
```
#Creating a 3D scatter plot using 'Scatter3d' with longitude, latitude, and
accessibility_index
data = go.Scatter3d(
    x=landmarks_df['longitude'],
    y=landmarks_df['latitude'],
    z=landmarks_df['accessibility_index'],
    mode='markers',
    marker=dict(
        size=5,
        color=landmarks_df['accessibility_index'],
        colorscale='Viridis',
        opacity=0.8
    ),
    text=landmarks_df['feature_name']
)

#Defining the layout of the plot with title and axis labels
layout = go.Layout(
    title='3D Visualization of Accessibility Index',
    scene=dict(
        xaxis=dict(title='Longitude'),
        yaxis=dict(title='Latitude'),
        zaxis=dict(title='Accessibility Index'),
    ),
)

#Updating the layout by adding camera settings and annotations
layout = go.Layout(
    title='3D Visualization of Accessibility Index',
    scene=dict(
        xaxis=dict(title='Longitude'),
        yaxis=dict(title='Latitude'),
        zaxis=dict(title='Accessibility Index'),
        camera=dict(
            eye=dict(x=1.87, y=0.88, z=-0.64)
        ),
        #Adding annotations for the center of accessibility in the plot
        annotations=[dict(
            x=landmarks_df['longitude'].mean(), #X-position of the
annotation (mean of longitude values)
            y=landmarks_df['latitude'].mean(), #Y-position of the
annotation (mean of latitude values)
            z=landmarks_df['accessibility_index'].mean(), #Z-position of
the annotation (mean of accessibility index values)
            text="Center of Accessibility",
            showarrow=True, #Show an arrow pointing to the annotated point
            arrowhead=2,
            ax=0, #X offset for the text position relative to the poin
            ay=-75 #Y offset for the text position relative to the point
        )]
    ),
)
#Creating the figure with the data and layout defined above and display it
```

```
fig = go.Figure(data=[data], layout=layout)
fig.show()
```

Key Elements:

Axes:
The x-axis represents longitude, the y-axis represents latitude, and the z-axis represents the Accessibility Index. The Accessibility Index reflects how easily accessible a landmark is, with higher values indicating greater accessibility.

Data Points:
Each point in the scatter plot represents a landmark, with its position determined by its geographical coordinates (latitude and longitude) and its accessibility index value. The points vary in color and size, likely corresponding to different accessibility levels, with higher points (in green, yellow) representing landmarks with greater accessibility.

Concentration of Accessibility:
The center of accessibility (indicated with the highest density of points and greater heights on the z-axis) is around latitude -37.8 and longitude 144.95, likely corresponding to Melbourne's central areas, where landmarks are more accessible.

Color Variation:
The color gradient, ranging from purple (lower values) to yellow (higher values), provides a clear visual cue of varying accessibility across landmarks. Purple points, being closer to the base of the z-axis, indicate landmarks with lower accessibility indexes, while yellow points at higher positions show landmarks with high accessibility.

Spatial Distribution:
The plot indicates that landmarks with higher accessibility indexes are mostly concentrated in the central areas, while outlying regions have points with lower accessibility. This pattern reflects the common scenario where urban centers typically have better accessibility due to higher density of transport and public services.

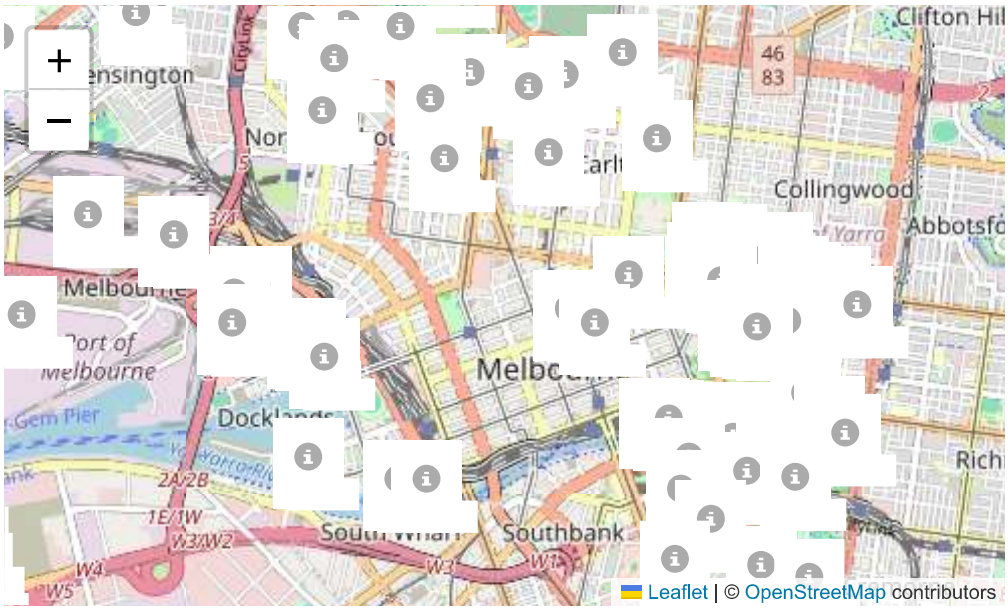
In [35]:
```
#Identifying poorly served clusters
poor_clusters = landmarks_df[landmarks_df['cluster'] == 0] #cluster 0
represents landmarks with lower accessibility

#Displaying landmarks needing improvement
print("Areas needing improvement:")
print(poor_clusters[['feature_name', 'latitude', 'longitude',
'accessibility_index']])

#Recommendations for new transport services
recommendations = poor_clusters[['feature_name', 'latitude',
'longitude']].copy()
recommendations.loc[:, 'proposed_improvement'] = 'New Bus Stop'
print("Recommended areas for new transport services:")
print(recommendations)
```

```
Areas needing improvement:
                                          feature_name   latitude
longitude  \
2                            St Mary's Anglican Church -37.803166
144.953762
5                                Greek Orthodox Church -37.808806
144.978259
9                                    Port of Melbourne -37.813738
144.917534
10                               Carlton Football Club -37.784086
144.961968
12                               Carlton Gardens North -37.801769
144.971998
..                                                ...        ...
...
227  Melbourne Wholesale Fruit, Vegetable & Flower ... -37.806873
144.923287
234                                   Westpac Centre -37.824246
144.979720
237                    Melbourne Girls Grammar School -37.831536
144.985089
239                                     David Jones -37.813313
144.964373
240                            Mercy Private Hospital -37.811897
144.984436

     accessibility_index
2            1738.556069
5            1446.152494
9             838.713053
10           1287.450239
12           1936.215085
..                   ...
227          1169.531303
234          1087.536716
237          1546.069269
239          2105.345341
```

```
240          1159.528357

[85 rows x 4 columns]
Recommended areas for new transport services:
                                            feature_name   latitude
longitude  \
2                             St Mary's Anglican Church -37.803166
144.953762
5                                 Greek Orthodox Church -37.808806
144.978259
9                                     Port of Melbourne -37.813738
144.917534
10                                Carlton Football Club -37.784086
144.961968
12                                Carlton Gardens North -37.801769
144.971998
..                                                  ...        ...
...
227  Melbourne Wholesale Fruit, Vegetable & Flower ... -37.806873
144.923287
234                                     Westpac Centre -37.824246
144.979720
237                      Melbourne Girls Grammar School -37.831536
144.985089
239                                       David Jones -37.813313
144.964373
240                               Mercy Private Hospital -37.811897
144.984436

    proposed_improvement
2           New Bus Stop
5           New Bus Stop
9           New Bus Stop
10          New Bus Stop
12          New Bus Stop
..                   ...
227         New Bus Stop
234         New Bus Stop
237         New Bus Stop
239         New Bus Stop
240         New Bus Stop

[85 rows x 4 columns]
```

In [52]:
```
recommendations.to_csv('transport_recommendations.csv', index=False)

#Visualizing the recommendations on a map using Folium
#Creating a map centered around the average location of the landmarks in
the poorly served cluster
map_center = [poor_clusters['latitude'].mean(),
poor_clusters['longitude'].mean()]
map = folium.Map(location=map_center, zoom_start=13)

#Adding a marker for each recommended area
for _, row in recommendations.iterrows():
    folium.Marker(
        location=[row['latitude'], row['longitude']],
        popup=f"{row['feature_name']} - {row['proposed_improvement']}",
        icon=folium.Icon(color='blue', icon='info-sign')
    ).add_to(map)

#Displaying the map
map
```

Out[52]:



My Recommendations

Based on the analysis of public transport accessibility to key landmarks in Melbourne, I have identified several areas that are currently underserved by public transport. These areas fall within the cluster identified as having the lowest accessibility. To improve connectivity and ensure that these landmarks are more easily accessible, I recommend the following:

Add New Bus Stops: For each of the landmarks in the poorly served cluster, I propose the installation of new bus stops. This will help bridge the gap in public transport coverage, making it easier for residents and visitors to reach these important places.

Focus on Specific Landmarks: Some of the landmarks that require immediate attention include St Mary's Anglican Church, Carlton Gardens North, and Melbourne Girls Grammar School, among others. By enhancing the public transport options near these locations, we can significantly improve their accessibility.

Monitor and Adjust: After implementing the new bus stops, I suggest closely monitoring the impact on accessibility. This will allow us to make any necessary adjustments, ensuring that the improvements meet the needs of the community effectively.

These recommendations are based on the current analysis and are aimed at enhancing public transport accessibility in a way that aligns with the city’s broader goals of improving connectivity and quality of life for its residents and visitors.

Policy Recommendations for Improving Public Transport Accessibility

Introduction

In this report, I aim to evaluate the adequacy of public transport services surrounding key places of interest in Melbourne. By analyzing the proximity and accessibility of train stations, tram stops, and bus stops to various landmarks, I have identified gaps in public transport coverage and proposed enhancements to improve overall connectivity.

Methodology

- **Data Collection**: I collected comprehensive data on landmarks, bus stops, tram stops, and train stations throughout Melbourne to build a detailed understanding of the existing transport infrastructure.
- **Feature Engineering**: I calculated the nearest distances from each landmark to the closest bus, tram, and train stops. Additionally, I developed an accessibility index to rate each landmark based on its level of public transport service.
- **Clustering Analysis**: To better understand the varying levels of accessibility, I used KMeans clustering to categorize landmarks into groups based on their accessibility profiles.

Findings

- **Cluster Analysis**: My analysis revealed three distinct clusters of landmarks based on their accessibility:
- **Cluster 0 (Purple)**: Landmarks with the lowest accessibility indices, indicating they are poorly served by public transport.
- **Cluster 1 (Green)**: Landmarks with moderate accessibility, where transport services are adequate but could be further improved.
- **Cluster 2 (Yellow)**: Landmarks with the highest accessibility, indicating these areas are well-served by public transport.

Recommendations

- **New Bus Stops**: I recommend establishing new bus stops at key locations within Cluster 0 to significantly improve accessibility for these underserved landmarks.
- **Enhance Existing Services**: For landmarks in Cluster 1, I propose upgrading existing transport services to further enhance accessibility, ensuring these areas meet the growing needs of the city.
- **Continuous Monitoring**: I suggest implementing a monitoring system to regularly assess transport accessibility across Melbourne. This will allow for timely improvements and ensure that the public transport network evolves in response to changing demands.

Conclusion

By following these recommendations, I believe the City of Melbourne can significantly enhance public transport connectivity, ensuring that all key places of interest are easily accessible to both residents and visitors. This will not only improve the quality of life in the city but also support sustainable growth and development.

Future Implementations

In [44]:
```
pedestrian_counts_df.head()
```

Out[44]:

	id	location_id	sensing_date	hourday	direction_1	di
0	35720220126	35	2022-01-26	7	196	20
1	521120220423	52	2022-04-23	11	262	21
2	49220230202	49	2023-02-02	2	22	21
3	45620230522	45	2023-05-22	6	36	39
4	1072020240601	107	2024-06-01	20	50	70

In [45]:
```
pedestrian_counts_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1856048 entries, 0 to 1856047
Data columns (total 11 columns):
 #   Column           Dtype
---  ------           -----
 0   id               int64
 1   location_id      int64
 2   sensing_date     datetime64[ns]
 3   hourday          int64
 4   direction_1      int64
 5   direction_2      int64
 6   pedestriancount  int64
 7   sensor_name      object
 8   location         object
 9   latitude         float64
 10  longitude        float64
dtypes: datetime64[ns](1), float64(2), int64(6), object(2)
memory usage: 155.8+ MB
```

In [46]:
```
#Converting sensing_date to datetime
pedestrian_counts_df['sensing_date'] =
pd.to_datetime(pedestrian_counts_df['sensing_date'])

#Separating latitude and longitude from the 'location' column
pedestrian_counts_df[['latitude', 'longitude']] =
pedestrian_counts_df['location'].str.split(', ', expand=True)
pedestrian_counts_df['latitude'] =
pedestrian_counts_df['latitude'].astype(float)
pedestrian_counts_df['longitude'] =
pedestrian_counts_df['longitude'].astype(float)

print(pedestrian_counts_df.head())
```

```
                id  location_id sensing_date  hourday  direction_1
direction_2  \
0    35720220126           35   2022-01-26        7          196
203
1   521120220423           52   2022-04-23       11          262
212
2    49220230202           49   2023-02-02        2           22
21
3    45620230522           45   2023-05-22        6           36
39
4  1072020240601          107   2024-06-01       20           50
70

   pedestriancount sensor_name                    location   latitude
\
0              399    SouthB_T  -37.82017828, 144.96508877 -37.820178
1              474    Eli263_T   -37.81252157, 144.9619401 -37.812522
2               43    Eli501_T  -37.80730068, 144.95956055 -37.807301
3               75    Swa148_T  -37.81414075, 144.96609379 -37.814141
4              120   280Will_T  -37.81246271, 144.95690188 -37.812463

    longitude
0  144.965089
1  144.961940
2  144.959561
3  144.966094
4  144.956902
```

In [47]:
```
#Converting tram stops DataFrame to GeoDataFrame
tram_stops_df['geometry'] = tram_stops_df.apply(lambda x:
Point((x.longitude, x.latitude)), axis=1)
tram_stops_gdf = gpd.GeoDataFrame(tram_stops_df, geometry='geometry')

#Converting train stations DataFrame to GeoDataFrame
train_stations_df['geometry'] = train_stations_df.apply(lambda x:
Point((x.longitude, x.latitude)), axis=1)
train_stops_gdf = gpd.GeoDataFrame(train_stations_df, geometry='geometry')
```

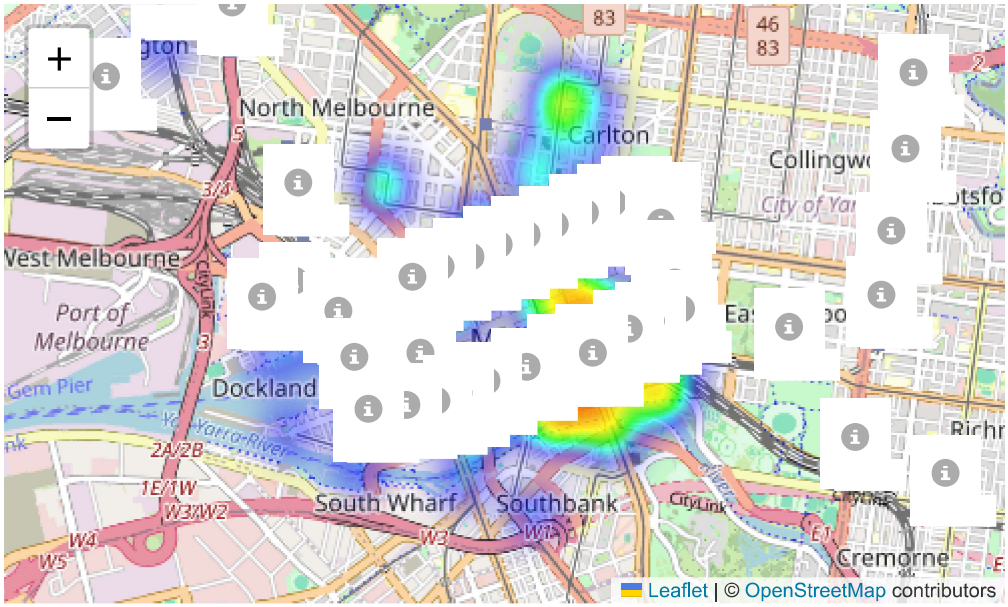
In [48]:
```
all_stops_gdf = pd.concat([tram_stops_gdf, train_stops_gdf])
```

In [49]:
```
#Converting to GeoDataFrame
pedestrian_gdf =
gpd.GeoDataFrame(pedestrian_counts_df,geometry=gpd.points_from_xy(pedestria
n_counts_df.longitude, pedestrian_counts_df.latitude))
```

In [50]:
```
#Spatial join between pedestrian counts and transport stops
joined_gdf = gpd.sjoin(pedestrian_gdf, all_stops_gdf, how='left',
predicate='intersects')

#Analyze the joined data
stop_traffic = joined_gdf.groupby('station')
['pedestriancount'].sum().reset_index()

#Define the threshold for high pedestrian traffic
threshold = 1000

#Identify stops with high pedestrian traffic
high_traffic_stops = stop_traffic[stop_traffic['pedestriancount'] >
threshold]
```

In [51]:
```
map_center = [pedestrian_gdf.geometry.y.mean(),
pedestrian_gdf.geometry.x.mean()]
m = folium.Map(location=map_center, zoom_start=13)

#Adding heatmap for pedestrian traffic
HeatMap(data=pedestrian_gdf[['latitude', 'longitude',
'pedestriancount']].values.tolist(), radius=15).add_to(m)

#Adding tram stops (blue markers)
for _, row in tram_stops_gdf.iterrows():
    folium.Marker(
        [row.geometry.y, row.geometry.x],
        popup=f"Tram Stop: {row['name']}",
        icon=folium.Icon(color='blue', icon='info-sign')
    ).add_to(m)

#Adding train stops (red markers)
for _, row in train_stops_gdf.iterrows():
    folium.Marker(
        [row.geometry.y, row.geometry.x],
        popup=f"Train Station: {row['station']}",
        icon=folium.Icon(color='red', icon='info-sign')
    ).add_to(m)
m
```

Out[51]:



This map provides a visual representation of the public transport infrastructure in Melbourne, focusing on train stations, tram stops, and pedestrian traffic. The blue markers are used to represent tram stops, while red markers denote train stations, allowing for a clear distinction between the two modes of transport.

To provide additional insights, I have included a heatmap layer that displays pedestrian activity across various locations in the city. The intensity of the heatmap varies, with brighter areas indicating higher concentrations of pedestrian traffic. This layer helps in identifying areas with significant foot traffic in proximity to public transport stops, allowing me to assess the adequacy of public transport services.

This visualization not only highlights the geographical distribution of tram and train stops but also shows the flow of pedestrians in the area.